Algorithms for partitioning well-clustered graphs

Luca Zanetti

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of Doctor of Philosophy in the Faculty of Engineering, Department of Computer Science.

October 2017

36000 words

Abstract

Graphs occurring in the real world usually exhibit a high level of order and organisation: higher concentration of edges within the same group of vertices, and lower concentration among different groups. A common way to analyse these graphs is to partition the vertex set of a graph into clusters according to some connectivity measure. Graph clustering has been widely applied to many fields of computer science, from machine learning to bioinformatics and social network analysis.

The focus of this thesis is to design and analyse algorithms for partitioning graphs presenting a strong cluster-structure, which we call well-clustered. We first study the spectral properties of the Laplacian matrix of such graphs, and prove a structure theorem that relates the eigenvectors corresponding to the smallest eigenvalues of the Laplacian matrix of a graph to the structure of its clusters.

We then harness this theorem to analyse Spectral Clustering, arguably the most popular graph clustering algorithm. We give for the first time approximation guarantees on the number of misclassified vertices by Spectral Clustering when applied to well-clustered graphs.

Since Spectral Clustering needs to compute as many eigenvectors of the Laplacian matrix as the number of clusters in the graph, its performance deteriorates as this number grows. We present an algorithm that overcomes this issue without compromising its accuracy. This algorithm runs in time nearly linear in the number of the edges and independently of the number of clusters in the input graph.

Finally, we tackle the problem of partitioning a graph whose description is distributed among many sites. We present a distributed algorithm that works in a few synchronous rounds, requires limited communication complexity, and achieves the same guarantees of Spectral Clustering as long as the clusters are balanced in size.

Acknowledgements

First of all I would like to thank my supervisor, He Sun, for his constant and never-ending support. For helping me bring order to the chaos of my proofs, writing, and presentations.

I would also like to thank my coauthor Richard Peng, for the many discussions that inspired several results in this thesis.

Special thanks go to the many friends I met during my PhD studies, first in Germany and then in the UK, for the laughs, pints, and conversations we had. They made these four years much more interesting and enjoyable. Thanks also to the old friends from my hometown whom I know I can always count on.

Finally, thanks to my parents, Oreste and Patrizia, and my sisters, Sonia and Cinzia (in rigorous order of age). I wouldn't be writing these acknowledgements without their material and moral support.

Declaration

I declare that the work in this dissertation was carried out in accordance with the requirements of the University's *Regulations and Code of Practice for Research Degree Programmes* and that it has not been submitted for any other academic award. Except where indicated by specific reference in the text, the work is the candidate's own work. Work done in collaboration with, or with the assistance of, others, is indicated as such. Any views expressed in the dissertation are those of the author.

SIGNED: ..... DATE: .....

Table of contents

List of figures

List of algorithms

Chapter 1

Introduction

The problem of grouping together objects that share similar characteristics, also called *clustering*, is one of the most important in computer science. When the relations between the objects are represented by a graph, this problem takes the name of *graph partitioning*. Informally, in graph partitioning we want to cut a graph in pieces (called *clusters*) so that, on average, vertices inside one piece are better connected to one another than to vertices in the rest of the graph.

Graph partitioning has countless practical applications [26]. Let us mention a few of these. Suppose, for example, our graph represents a social network: vertices are users of the social network, and edges represent relations of friendship between pairs of users. Then, graph partitioning corresponds to finding *communities* of users, i.e., groups of users with strong social ties. Suppose now our graph encodes the interactions between different proteins in a cell. Graph partitioning can be used to find groups of proteins that perform similar functions in the cell. Finally, consider the problem of identifying different objects in a picture, which is known as image segmentation. This problem can be formalised as a graph partitioning problem in the following way: we construct a graph whose vertices represent the pixels in the image and between any two we place an edge with a weight that depends on how far the corresponding pixels are located in the image and how similar in colour are these two pixels. This is called the *similarity graph* of the image. Partitioning a similarity graph has been shown to achieve very good image segmentation results [60, 70].

One way to formalise the problem of graph partitioning is via the notion of *conductance* [62]: given a set of vertices S, the conductance of S is the number of edges between S and the rest of the graph, divided by the number of edges with at least one endpoint in S. A subset of vertices with low conductance corresponds to a subset

of vertices that have, on average, more connections between one another than to the rest of the graph. They form, in other words, a cluster. We can then rephrase the problem of partitioning a graph into k clusters as the problem of finding k subsets of low conductance in the graph. Such formalisation, however, gives rise to an NP-hard problem [61] for which the best known approximation algorithm can only achieve an $O(\sqrt{\log n})$-approximation ratio [8].

Despite their worst-case hardness, graph partitioning problems are routinely solved in practice. Arguably, one of the most famous heuristics for graph partitioning is *Spectral Clustering* [74]. Spectral Clustering works by first computing the bottom k eigenvectors of the Laplacian matrix of the graph. These eigenvectors are then used to embed the vertices in a k-dimensional Euclidean space, which is called the *spectral embedding* of the graph. The points in this embedding are then partitioned with a geometric clustering algorithm such as k-means. Finally, the corresponding partition of the graph is returned.

Thanks to its simplicity, low running time, and effectiveness in tackling real-world instances, Spectral Clustering has enjoyed widespread popularity since its conception. A theoretical justification for why Spectral Clustering often outputs an almost-optimal partitioning can be found in the Cheeger [6, 62] and higher-order Cheeger inequalities [42, 39]. This justification, however, falls well short of providing a rigorous analysis of Spectral Clustering. Indeed, known analysis of its approximation guarantees are limited to stochastic models [58] or toy examples [50], and a strong theoretical understanding is still missing.

In this thesis we try to advance our understanding of Spectral Clustering, and attempt to give an answer to the question of why Spectral Clustering performs so well in practice. Our starting point is the definition of a new class of graphs, which we call *well-clustered*, that are characterised by a well-defined cluster-structure that can be recovered by Spectral Clustering. To the best of our knowledge, this is the first attempt in literature to provide *reasonable* sufficient conditions for Spectral Clustering to work, at least outside of toy examples and stochastic models.

Our investigation on the power of Spectral Clustering is built upon a theorem that relates the *shape* of the eigenvectors of the Laplacian matrix of a well-clustered graph to the structure of its clusters. Recall that the Laplacian matrix of an n-vertex graph G is the $n \times n$ matrix $\mathcal{L}_G = \mathbb{I} - D_G^{-1/2} A_G D^{-1/2}$, where $\mathbb{I}$ is the identity matrix, D_G is a diagonal matrix filled with the degrees of the vertices, and A_G is the adjacency matrix of G. Given a subset of vertices S, we call a vector χ_S an indicator vector for S if it

is equal to one on S and zero everywhere else. A fundamental fact in spectral graph theory states that a vector χ is an eigenvector of eigenvalue zero for $\mathcal{L}_G$ if and only if χ can be expressed as a linear combination of the indicator vectors of the connected components of G [17]. Our structure theorem is a robust version of this fact[1]: let G be a well-clustered graph with optimal clusters $S_1, \ldots, S_k$ and let $\chi_{S_1}, \ldots, \chi_{S_k}$ be the indicator vectors of such clusters. Then, the k bottom eigenvectors of $\mathcal{L}_G$ can be approximately expressed as linear combinations of $\chi_{S_1}, \ldots, \chi_{S_k}$.

Thanks to this structure theorem we are able to show that Spectral Clustering is able to approximately recover the optimal clusters of a well-clustered graph. Remarkably, we can show that even clusters of small size are well-approximated by Spectral Clustering, which is usually a very challenging problem.

The efficiency of Spectral Clustering crucially depends on the number of clusters into which the graph needs to be partitioned. In particular, when the number of clusters k is large, there are two issues that need to be addressed: first of all, we need to compute the k bottom eigenvectors of the Laplacian, which necessarily takes $\Omega(n \cdot k)$ time on a graph of n vertices. Secondly, we need to solve a k-means problem, which again usually takes at least $\Omega(n \cdot k)$ time. Therefore, when k is large, Spectral Clustering becomes much less efficient. To overcome this drawback, we present an algorithm for partitioning well-clustered graphs whose running time is nearly-linear in the number of edges of the graph and independent of k. Instead of computing the spectral embedding, our algorithm uses an embedding based on the heat-kernel of the graph [59], which can be computed quickly by combining fast matrix-exponential algorithms [51] with Johnson-Lindestrauss random projections [31]. Moreover, our algorithm exploits the geometry of the heat-kernel embedding to partition points in the embedding without having to resort to standard k-means algorithms.

The final part of the thesis is devoted to the design and analysis of a distributed algorithm for partitioning a well-clustered graph. More precisely, we assume we have a distributed network in which each node is a computational unit and communication can occur only between neighbouring nodes. Our goal is to assign a label to each node of the network that indicates which cluster the node belongs to. We present a distributed algorithm that, assuming the topology of the network can be described by a well-clustered graphs with clusters of balanced size, computes an almost-optimal clustering. In other words, we are able to partition a network in a distributed way with almost the same guarantees of Spectral Clustering. Moreover, for reasonable

[1]For simplicity here we state the version of the theorem for *regular* graphs only.

network topologies, our algorithm requires only $O(\text{poly}\log n)$ synchronous rounds and $O(n \cdot \text{poly}\log n)$ communication complexity. It is based on two novel ingredients:
(1) a distributed sparsification procedure which, given as input a dense graph, outputs a sparse subgraph with the same cluster-structure as the original graph, and whose purpose is to considerably reduce the communication complexity of the overall algorithm.
(2) A distributed clustering algorithm which is inspired by recent developments in local algorithms for graph clustering [5, 65] and whose analysis crucially depends on the aforementioned structure theorem of well-clustered graphs. Apart from giving theoretical guarantees for this algorithm, we also provide experimental results which show the effectiveness of our sparsification procedure in reducing the size of a graph without changing its cluster-structure. We believe this procedure can be very useful to speed-up Spectral Clustering in various applications.

1.1 Organisation of the thesis

The thesis is organised into the following chapters.

Chapter 2 provides the necessary background to understand the rest of the thesis. After a review of the basic definitions and notation about graphs and matrices, the chapter contains a short and self-contained introduction to spectral graph theory and graph partitioning.

Chapter 3 contains the formal definition of well-clustered graphs and a comparison between this definition and other formalisations proposed in literature. The main result of this chapter is a theorem that relates the structure of the bottom eigenvectors of the Laplacian matrix of a graph to the structure of its clusters. To analyse the strengths and weaknesses of this theorem, we also discuss standard matrix-perturbation theorems and their application to the analysis of stochastic models of graphs. This chapter is mostly based on Section 3 of [56], which is a joint work with Richard Peng and He Sun, and whose preliminary version appeared in [55].

Chapter 4 is mainly devoted to the analysis of Spectral Clustering for well-clustered graphs. For this purpose, we prove several interesting facts about the spectral embedding of well-clustered graphs. We harness these properties to analyse the performance of Spectral Clustering on well-clustered graphs. The results presented in this chapter first appeared in Section 4 of [56].

Chapter 5 presents our nearly-linear time algorithm for partitioning well-clustered graphs. The analysis of the algorithm consists of two main ingredients: we first show that the heat-kernel embedding approximates the spectral embedding in well-clustered graphs. Secondly, we develop a linear time algorithm to compute an almost optimal k-means clustering of such an embedding. We end the chapter with a discussion about the power of the heat-kernel for graph partitioning, showing an example of a graph where spectral methods fail but a method based on the heat-kernel succeeds. This chapter is based on Section 5 of [56].

Chapter 6 deals with graph partitioning in the distributed setting. We present an algorithm to partition a distributed network that achieves almost the same guarantees as a sequential implementation of Spectral Clustering. Moreover, the algorithm requires a small number of synchronous rounds and limited communication complexity. One key ingredient of this algorithm is a distributed sparsification procedure that reduces the number of edges of a graph while preserving its cluster structure. We present experimental results validating the theoretical guarantees of this procedure on real-world instances. The chapter ends by discussing several open problems for distributed graph partitioning. The results in this chapter are based on [67], which is a joint work with He Sun, and whose preliminary version appeared in [68].

Chapter 2

Preliminaries

In this chapter we review basic concepts about graphs and matrices that will be used in the rest of this thesis. Apart from reminding the reader of the necessary background, this chapter serves as an introduction to key notation and concepts. It is organised as follows: in Section 2.1 we introduce definitions and notation about graphs. In Section 2.2 we recall basic concepts about linear algebra. In Section 2.3 we review the basics of spectral graph theory, while Section 2.4 is devoted to a brief introduction to graph partitioning with an emphasis on spectral methods. The last section of this chapter contains additional notation used in this thesis.

2.1 Graphs

A graph is a collection of vertices connected by edges. We represent a graph G as a pair (V, E), where V is a set of vertices and $E \subseteq \binom{V}{2}$ is a set of edges. We usually denote with n the number of vertices and with m the number of edges. Since graphs are useful, for example, to represent similarities between objects, it is sometimes necessary to express the strength of a link between two objects. For this reason, we will often work with a weighted graph $G = (V, E, w)$, where $w : V \times V \to \mathbb{R}_{\geq 0}$ is a weight function and $w(u, v)$ represents the weight of the edge between vertices u and v. If there is no edge between u and v, $w(u, v) = 0$. We only consider undirected graphs, i.e., graphs for which $w(u, v) = w(v, u)$ for any $\{u, v\} \in E$. If $w(u, v) = 1$ for any $\{u, v\} \in E$, then we say that G is unweighted, and we often drop the function w from the definition of G.

The *degree* d_v of a vertex v is the sum of the weights of the edges having v as one of their endpoints, i.e., $d_v = \sum_{\{u,v\} \in E} w(u, v)$. Given a subset of vertices $S \subseteq V$, the volume of S is the sum of the degrees of the vertices in S: $\mathrm{vol}(S) = \sum_{v \in S} d_v$. The

cut-value between two disjoint subsets of vertices $S, T \subset V$ is the sum of the weights of the edges between S and T, i.e.,

$$w(S,T) = \sum_{\substack{\{u,v\}\in E \\ u\in S, v\in T}} w(u,v).$$

To indicate the edges between a subset of vertices S and the rest of the graph we sometimes use the following shorthand: $\partial S = E(S, V \setminus S)$.

A graph G is said to be *connected* if any two vertices u and v are connected by a path through the edges of the graph, otherwise it is called *disconnected*. A maximal subset of vertices in which any two vertices are connected by a path is called a *connected component*.

2.2 Matrices

This thesis focuses on linear algebraic approaches to graph partitioning. For this reason, we recall in this section some basic definitions and properties of vectors and matrices. This is far from a complete introduction to linear algebra and we refer the reader to [29] for a more exhaustive treatment of the topic.

We limit ourselves to consider (column) vectors in Euclidean spaces. Let n be the dimension of the Euclidean space considered. We express a vector $v \in \mathbb{R}^n$ as $v = (v_1, \ldots, v_n)^\intercal$. Given two vectors $u, v \in \mathbb{R}^n$, their *inner product* is defined as $\langle u, v\rangle = u^\intercal v = \sum_{i=1}^n u_i \cdot v_i$ and we say that u, v are *orthogonal*, and we write $u \perp v$, if $\langle u, v\rangle = 0$. The *norm* of v is defined as $\|v\| = \sqrt{\langle v, v\rangle}$.

A *subspace* $S \subseteq \mathbb{R}^n$ is a set of vectors closed under scalar multiplication and addition. A *linear combination* of vectors is an expression of the form $c_1 v_1 + \cdots + c_k v_k$, where $v_1, \ldots, v_k \in \mathbb{R}^n$ and $c_1, \ldots, c_k \in \mathbb{R}$. The set of all linear combinations of $v_1, \ldots, v_k \in \mathbb{R}^n$ forms a subspace, and is denoted by

$$\operatorname{span}\{v_1, \ldots, v_k\} = \{c_1 v_1 + \cdots + c_k v_k : c_1, \ldots, c_k \in \mathbb{R}\}.$$

We say vectors $v_1, \ldots, v_k \in \mathbb{R}^n$ are *linearly dependent* if there exist scalars $c_1, \ldots, c_k \in \mathbb{R}$, not all zero, such that $c_1 v_1 + \cdots + c_k v_k = 0$; otherwise we say they are *linearly independent*. If $S = \operatorname{span}\{v_1, \ldots, v_k\}$ and $v_1, \ldots, v_k \in \mathbb{R}^n$ are linearly independent, we say that $\{v_1, \ldots, v_k\}$ is a *basis* for S. Moreover, if these vectors $v_1, \ldots, v_k$ are pairwise orthogonal and have all norm one, then they form an *orthonormal* basis. The dimension

of a subspace S, denoted by $\dim S$, is the number of vectors in a basis for S. Finally, we say that two subspaces $S, T \subseteq \mathbb{R}^n$ are orthogonal, and write $S \perp T$, if for any $x \in S$ and $y \in T$ we have that $x \perp y$.

A *matrix* $A \in \mathbb{R}^{m \times n}$ is an m-by-n array of real numbers $(A_{ij})_{\substack{i=1,\dots,m \\ j=1,\dots,n}}$. Its transpose $A^\intercal \in \mathbb{R}^{n \times m}$ is the matrix obtained by swapping the rows and columns of A, that is, $A^\intercal_{ij} = A_{ji}$. We say that A is *symmetric* if $A = A^\intercal$. The *image* of A is defined as $\operatorname{im} A = \{y \in \mathbb{R}^m : \exists x \in \mathbb{R}^n \setminus \{0\} \text{ s.t. } Ax = y\}$, and its *kernel* is $\ker A = \{x \in \mathbb{R}^n : Ax = 0\}$. Notice that $\operatorname{im} A$ is a subspace of $\mathbb{R}^m$, while $\ker A$ is a subspace of $\mathbb{R}^n$. The dimension of the image is called the rank of A , and it is equal to the number of linearly independent columns (or rows) of A, while the dimension of the kernel is called nullity. They are related by the *rank-nullity theorem*:

$$\dim \operatorname{im} A + \dim \ker A = n.$$

If A is a square matrix, i.e., $m = n$, and symmetric, then $\ker A = \mathbb{R}^n \setminus \operatorname{im} A$, and $\ker A$ and $\operatorname{im} A$ are orthogonal to each other.

Now we introduce the notion of eigenvalues and eigenvectors. Let $A \in \mathbb{R}^{m \times n}$. If $\lambda \in \mathbb{R}$ and $v \in \mathbb{R}^n \setminus \{0\}$ satisfy

$$Av = \lambda v,$$

then λ is an *eigenvalue* of A with corresponding *eigenvector* v. The set of all eigenvalues of A is called the *spectrum* of A, and it is denoted by $\sigma(A)$.

When A is not symmetric it is sometimes more useful to consider *singular values* instead of eigenvalues. We say that s is a singular value of A if and only if s is the non-negative square root of an eigenvalue of $A^\intercal A$. It can be shown that A and $A^\intercal$ have the same nonzero singular values, and $A^\intercal A$ and $AA^\intercal$ have the same nonzero eigenvalues.

For the rest of the section we will assume that A is an $n \times n$ symmetric matrix. In this case, the *spectral theorem* says A has n eigenvalues and there exists an orthonormal basis of $\mathbb{R}^n$ that consists of eigenvectors of A. Another way to state this theorem is that A can be decomposed as

$$A = \sum_{i=1}^{n} \lambda_i f_i f_i^\intercal,$$

where $\lambda_1, \dots, \lambda_n$ are the eigenvalues of A and $f_1, \dots, f_n$ are the corresponding orthonormal eigenvectors.

Thanks to this decomposition we can generalise real-valued functions to matrix-valued functions: given a real-valued function $g : \mathcal{D} \to \mathbb{R}$ with domain $\mathcal{D} \subseteq \mathbb{R}$ and a symmetric matrix $A \in \mathbb{R}^{n\times n}$ with eigendecomposition $A = \sum_{i=1}^{n} \lambda_i f_i f_i^\intercal$ and $\sigma(A) \subseteq \mathcal{D}$, the matrix $g(A) \in \mathbb{R}^{n\times n}$ is defined as

$$g(A) \triangleq \sum_{i=1}^{n} g(\lambda_i) f_i f_i^\intercal.$$

Notice that $\sigma(A) \subseteq \mathcal{D}$ is needed for our definition to make any sense.

An example of matrix-valued functions is the matrix-power, which takes a matrix A and output the product of A with itself $k \in \mathbb{N}$ times: if A is symmetric, it is the same as computing

$$A^k = \sum_{i=1}^{n} \lambda_i^k f_i f_i^\intercal.$$

Based on the formula above, we can define A^k for an arbitrary $k \in \mathbb{R}$. Another notable example is the matrix-exponential $\exp(A) = \sum_{k=0}^{\infty} A^k/k!$, which can be written as $\exp(A) = \sum_{i=1}^{n} \exp(\lambda_i) f_i f_i^\intercal$. Notice that matrix-valued functions do not necessarily share all the properties of their real-valued analogue. For instance, we know that for any $x, y \in \mathbb{R}$, $\exp(xy) = \exp(x)\exp(x)$, but $\exp(AB)$ is not always equal to $\exp(A)\exp(B)$. Indeed, it is equal if and only if $AB = BA$.

The *inverse* of A, if it exists, is the unique matrix A^{-1} such that $AA^{-1} = A^{-1}A = \mathbb{I}$, where $\mathbb{I}$ is the identity matrix. By the previous discussion, A^{-1} can be decomposed as $A^{-1} = \sum_{i=1}^{n} \lambda_i^{-1} f_i f_i^\intercal$. It is clear, then, A^{-1} exists if and only if $0 \notin \sigma(A)$.

Sometimes it is useful to consider a generalisation of the inverse of a matrix with zeroes in its spectrum. For this reason we introduce the *pseudoinverse* $A^\dagger$. This matrix has the same kernel as A and acts as its inverse on the image of A. It can be decomposed as

$$A^\dagger = \sum_{\lambda_i \neq 0} \frac{1}{\lambda_i} f_i f_i^\intercal.$$

The matrix $AA^\dagger = A^\dagger A = \sum_{\lambda_i \neq 0} f_i f_i^\intercal$ can be seen as the projection on the subspace spanned by the eigenvectors corresponding to the nonzero eigenvalues of A, i.e., the image of A. More generally, given a subspace S with orthonormal basis $\{v_1, \ldots, v_k\}$, the projection on S is the matrix defined as $P_S = \sum_{i=1}^{k} v_i v_i^\intercal$. Observe that P_S acts as the identity matrix on S, and as the all-zeroes matrix on its orthogonal subspace.

There are several characterisations of the eigenvalues of real symmetric matrices. These characterisations reduce the task of finding the eigenvalues of a matrix to a

convex optimisation problem. Let A be a matrix with eigenvalues $\lambda_1 \leq \cdots \leq \lambda_n$ and corresponding eigenvectors $f_1, \ldots, f_n$. The Rayleigh quotient theorem states that

$$\lambda_i = \min_{\substack{x \in \mathbb{R}^n \setminus \{0\} \\ x \perp f_1, \ldots, f_{i-1}}} \frac{x^\intercal A x}{x^\intercal x} = \max_{\substack{x \in \mathbb{R}^n \setminus \{0\} \\ x \perp f_{i+1}, \ldots, f_n}} \frac{x^\intercal A x}{x^\intercal x}, \tag{2.1}$$

and the corresponding eigenvector f_i is the minimiser (maximiser) of this optimisation problem. Another useful characterisation of the eigenvalues of real symmetric matrices is the Courant-Fisher theorem, which states that

$$\lambda_i = \min_{S \colon \dim S = i} \max_{x \in S \setminus \{0\}} \frac{x^\intercal A x}{x^\intercal x} = \max_{S \colon \dim S = n-i+1} \min_{x \in S \setminus \{0\}} \frac{x^\intercal A x}{x^\intercal x}, \tag{2.2}$$

where S is a subspace of $\mathbb{R}^n$.

The spectral norm of a symmetric matrix A, denoted by $\|A\|$, is equal to the largest eigenvalue of A in absolute value: $\|A\| = \max\{|\lambda| : \lambda \in \sigma(A)\}$. We say that A is positive-semidefinite (PSD) if all the eigenvalues of A are non-negative.

Finally, the trace of a square symmetric matrix $A \in \mathbb{R}^{n \times n}$, denoted by $\operatorname{tr} A$, is the sum of its diagonal entries, i.e., $\operatorname{tr} A = \sum_{i=1}^n A_{i,i}$. It is also equal to the sum of all the eigenvalues of A.

2.3 Graphs and matrices

Spectral graph theory relates combinatorial properties of a graph to the spectrum of some notable matrices. Spectral graph theory has a long history. Gustav Kirchhoff, with his matrix tree theorem, can be seen as a forefather of the field: he proved a formula that, in modern terms, relates the number of spanning trees in a graph to the eigenvalues of its Laplacian matrix [36]. But it was only since the end of the 1980s that spectral graph theory has been fully established and started to have a prominent role in combinatorics, theoretical computer science, and machine learning. In particular, the last decade has seen a flurry of results demonstrating the power and breadth of spectral techniques in algorithm design, which has been christened as "the Laplacian paradigm" [69]. The aim of this section is to define the basic matrices related to a graph, and these matrices will be our object of study in later chapters.

Given an undirected graph $G = (V, E, w)$ of $|V| = n$ vertices, its *adjacency matrix* A_G is the $n \times n$ symmetric matrix such that $A_G(u, v) = w(u, v)$. From the spectrum of A_G we can already learn various interesting combinatorial properties of G, such as the

number of connected components and if G is bipartite [17]. Another notable matrix is the *random-walk matrix* of G, which is the transition matrix of a 1-step random walk in the graph and is defined as $P_G \triangleq D_G^{-1} A_G$, where $D_G \in \mathbb{R}^{n \times n}$ is a diagonal matrix filled with the degrees of the vertices, i.e., $D_G(u,u) = d_u$. In fact, $P_G(u,v) = w(u,v)/d_u$ is the probability that we move from u to its neighbour v with probability proportional to the weight $w(u,v)$. Like the adjacency matrix, the spectrum of P_G encodes several combinatorial properties of the graph. Unlike the adjacency matrix, however, the random-walk matrix is not symmetric. For this reason, it is usually more convenient to work with the *Laplacian* matrix of G, denoted by $\mathcal{L}_G$.[1]

Definition 2.1. Let $G = (V, E, w)$ be a weighted graph of n vertices. The Laplacian of G is the $n \times n$ matrix

$$\mathcal{L}_G \triangleq \mathbb{I} - D_G^{-1/2} A_G D_G^{-1/2}.$$

From the definition above it is clear the Laplacian is always symmetric. Moreover, the spectra of P_G and $\mathcal{L}_G$ are closely related: λ is an eigenvalue of P_G with eigenvector f if and only if $1 - \lambda$ is an eigenvalue of $\mathcal{L}_G$ with corresponding eigenvector $D^{1/2} f$. Therefore, $\mathcal{L}_G$ preserves all the spectral properties of P_G.

Another interesting property of the Laplacian is that it can be decomposed in a sum of $|E|$ rank-1 matrices as follows: for any edge $e = \{u, v\}$, we define a vector c_e as

$$c_e(z) = \begin{cases} 1/\sqrt{d_u} & \text{if } z = u \\ -1/\sqrt{d_v} & \text{if } z = v \\ 0 & \text{otherwise} \end{cases}$$

where we choose the role of u and v, i.e., which entry will be positive and which one negative, arbitrarily. Then, we can decompose $\mathcal{L}_G$ as

$$\mathcal{L}_G = \sum_{e=\{u,v\} \in E} w(u,v) \cdot c_e c_e^\intercal.$$

Of extreme importance in spectral graph theory are the quadratic forms of the Laplacian. For any nonzero $f \in \mathbb{R}^n$, we define the Rayleigh quotient of f with respect to G as

$$\mathcal{R}_G(f) \triangleq \frac{f^\intercal D_G^{1/2} \mathcal{L}_G D_G^{1/2} f}{f^\intercal D_G f}, \tag{2.3}$$

[1] In many textbooks $\mathcal{L}_G$ is called the *normalised* Laplacian of G.

and if we fix $g = D_G^{1/2} f$, the following relations hold:

$$\mathcal{R}_G(f) = \frac{g^\intercal \mathcal{L}_G g}{g^\intercal g} = \frac{\sum_{\{u,v\}\in E} w(u,v)\,(f(u) - f(v))^2}{\sum_{v\in V} d_v f(v)^2}. \tag{2.4}$$

From (2.4) and the min-max characterisation of eigenvalues (2.2) we can learn several interesting properties about the spectrum of $\mathcal{L}_G$:

- The matrix $\mathcal{L}_G$ is positive-semidefinite. This follows from the fact that, since edge-weights are always nonnegative, (2.4) is always nonnegative as well.
- The number of zero eigenvalues is equal to the number of connected components of G. To see this, consider a nonzero $g = D_G^{1/2} f$. For $g^\intercal \mathcal{L}_G g$ to be zero, from (2.4) we can deduce that f must be constant on the connected components of G. It is not difficult to show that the maximal number of orthogonal vectors satisfying this constraint is equal to the number of connected components of the graph. As a corollary, this implies that $\mathcal{L}_G$ has exactly one trivial eigenvalue with eigenvector $D_G^{1/2} 1$ when G is connected.
- The largest eigenvalue of $\mathcal{L}_G$ is at most 2, and it is equal to 2 if and only if one of G's connected components is bipartite. To sketch a proof of this fact, notice that (2.4) achieves its maximum if $f(u)$ and $f(v)$ have opposite sign for every edge $\{u, v\} \in E$. This can happen if and only if one of G's connected components is bipartite.

For other basic facts about the spectrum of $\mathcal{L}_G$ we refer the reader to [17]. In the rest of this thesis, unless stated otherwise, we use $\lambda_1 \leq \cdots \leq \lambda_n$ to indicate the eigenvalues of the Laplacian in non-decreasing order, and $f_1, \ldots, f_n$ for the corresponding eigenvectors.

2.4 Spectral methods for graph partitioning

One of the most important applications of spectral methods in computer science is graph partitioning, which requires cutting a graph in pieces so that each piece is better connected on the inside than towards the outside. There are several different ways to formulate the problem of graph partitioning, each one tailored to different application scenarios. For example, in the *edge expansion problem* the objective is to minimise the number of edges we have to remove to separate a subset of vertices of large volume from the rest of the graph. This problem can be reformulated as the

problem of finding a subset of vertices minimising the *conductance* of the graph. Given a graph $G = (V, E, w)$ and a nonempty subset of vertices $S \subset V$, the conductance of S, denoted by $\phi_G(S)$, is defined as the ratio between the total weight of the edges with exactly one endpoint in S and the minimum between the volume of S and the volume of its complement $V \setminus S$. In formula,

$$\phi_G(S) \triangleq \frac{w(S, V \setminus S)}{\min\{\mathrm{vol}(S), \mathrm{vol}(V \setminus S)\}}. \tag{2.5}$$

Notice that $0 \leq \phi_G(S) \leq 1$ for any set $S \subset V$, and $\phi_G(S) = 0$ if and only if S is a connected component of the graph. In general, $\phi_G(S)$ represents how "well connected" is S to the rest of the graph.

The conductance of the whole graph G is defined as the minimum conductance over all nonempty subsets of vertices $S \subset V$, that is,

$$\phi_G \triangleq \min_{S \subset V, S \neq \emptyset} \phi_G(S). \tag{2.6}$$

Finding a set S of vertices minimising (2.6) has many practical and theoretical applications. For instance, if G represents a traffic network, the cut between such set S and $V \setminus S$ corresponds to the most significant bottleneck in the network. It corresponds, for example, in finding the greatest bottleneck in a network. Exactly computing the conductance of a graph, however, is NP-hard in the worst-case [47], and the current best approximation algorithm achieves only an $O(\sqrt{\log n})$ approximation ratio [8].

The celebrated discrete Cheeger inequality relates the second smallest eigenvalue λ_2 of $\mathcal{L}_G$, to the conductance ϕ_G of G, and can be seen as a quantitative version of the aforementioned statement that $\lambda_2 = 0$ if and only if G is disconnected. Cheeger inequality was first proved in 1969 in the context of Riemannian geometry [14] and transferred to the discrete world by [6, 22, 62] in the 1980s. The discrete Cheeger inequality states that

$$\frac{\lambda_2}{2} \leq \phi_G \leq \sqrt{2\lambda_2}. \tag{2.7}$$

This inequality is essentially tight, in the sense that there are infinite families of graphs achieving either direction up to a constant. For example, consider the n-dimensional hypercube H_n. The minimum conductance is achieved by one of the cuts dividing H_n in two $(n-1)$-dimensional hypercubes (there are n of these cuts). A simple computation reveals that $\phi_{H_n} = 1/n$, while the second smallest eigenvalue $\lambda_2(\mathcal{L}_{H_n}) = 2/n$ (see, e.g., [17]). Therefore, $\phi_{H_n} = \lambda_2(\mathcal{L}_{H_n})/2$, showing that the first inequality in (2.7)

is tight. For the second inequality let us consider the n-cycle C_n. The minimum conductance ϕ_{C_n} is achieved by a cut dividing C_n in two paths of $\lceil n/2 \rceil$ and $\lfloor n/2 \rfloor$ vertices, and therefore $\phi_{C_n} = 1/\lfloor n/2 \rfloor = \Theta(1/n)$. On the other hand, it can be shown that the the second smallest eigenvalue $\lambda_2(\mathcal{L}_{C_n}) = 1 - \cos(2\pi/n) = \Theta(1/n^2)$ (see, e.g., [17]), which implies that $\phi_{C_n} = \Theta\left(\sqrt{\lambda_2(\mathcal{L}_{C_n})}\right)$ and the second inequality in (2.7) is tight up to a constant.

Cheeger inequality not only gives us an approximation of the conductance of a graph, but an algorithm to find a partition achieving this approximation as well. The algorithm works as follows: first compute an eigenvector f_2 corresponding to the second smallest eigenvalue of $\mathcal{L}_G$; given a threshold $t \in \mathbb{R}$, define the set $S_t = \{u : f_2(u) \leq t \cdot \sqrt{d_u}\}$.The proof of Cheeger inequality guarantees there exists a threshold t such that $\phi_G(S_t) \leq \sqrt{2\lambda_2}$. Notice that there exist only n distinct sets S_t, so the algorithm can be implemented in polynomial time (see Algorithm 1 for a formal description of the algorithm). The guarantees on the approximation ratio of this algorithm given by Cheeger inequality (2.7), however, are good only if ϕ_G is large, e.g., $\phi_G = \Omega(1)$, but poor if ϕ_G is small, e.g., for the cycle C_n[2].

Algorithm 1 Spectral partitioning

input: a graph G
output: a set $S^\star$ satisfying (2.7)
Compute $f_2 \in \mathbb{R}^n$ such that $\mathcal{L}_G f_2 = \lambda_2 f_2$.
Order the vertices of G such that $f_2(v_1)/\sqrt{d_{v_1}} \leq \cdots \leq f_2(v_n)/\sqrt{d_{v_n}}$.
$\phi_0 = 1$
$S^\star = \emptyset$
for $i = 1, \ldots, n-1$ **do**
    $S = \{v_1, \ldots, v_i\}$
    if $\phi_G(S) \leq \phi_{i-1}$ **then**
        $\phi_i = \phi_G(S)$
        $S^\star = S$
return $S^\star$

To provide some intuition on the relation between λ_2 and ϕ_G, let's consider a subset of vertices S such that $0 < \mathrm{vol}(S) \leq \mathrm{vol}(V)/2$ with indicator vector $\chi_S \in \{0,1\}^n$ of S, i.e., a boolean vector such that $\chi_S(u) = 1$ if and only if $u \in S$. Then, by (2.4) we have that

$$\mathcal{R}_G(\chi_S) = \frac{\sum_{u \in S, v \notin S} w(u,v)}{\sum_{v \in S} d_v} = \frac{w(S, V \setminus S)}{\mathrm{vol}(S)} = \phi_G(S). \tag{2.8}$$

[2]Algorithm 1 actually returns a set of low conductance for C_n, but the guarantees given by (2.7) are bad.

Therefore, finding a set with minimum conductance in G corresponds to minimising $\mathcal{R}_G(\chi_S)$ over the set of vectors χ_S with $0 < \text{vol}(S) \leq \text{vol}(V)/2$. Thus, from (2.1) we can see that computing λ_2 corresponds to solving a convex relaxation for the conductance (2.6), and the Cheeger inequality gives approximation guarantees for the corresponding rounding algorithm. For a full proof of Cheeger inequality we refer the reader to [17]. We also mention that the idea of using the second bottom eigenvector f_2 of the Laplacian to partition a graph predates the formulation of the discrete Cheeger inequality and can be found, for example, in the foundational work of Fiedler [24, 25].

Finally, we generalise the notion of conductance to k-way partitions of a graph G. We call a collection of k subsets $A_1, \ldots, A_k \subset V$ a k-way partition of V if $\bigcup_{i=1}^{k} A_i = V$ and $A_i \cap A_j = \emptyset$ for $i \neq j$. Generalising the definition of conductance, we define the *k-way expansion constant* as

$$\rho_G(k) = \min_{\substack{S_1,\ldots,S_k \\ \text{partition of } V}} \max_{1\leq i\leq k} \phi_G(S_i), \tag{2.9}$$

Notice that when $k = 2$, $\rho_G(k)$ is just the conductance of the graph. It is then natural to ask if there exists a relation between $\rho_G(k)$ and λ_k, which corresponds to the Cheeger inequality (2.7) when $k = 2$. A positive answer has been recently provided by the *higher order Cheeger inequality* [39], which states that

$$\frac{\lambda_k}{2} \leq \rho_G(k) \leq O(k^3)\sqrt{\lambda_k}. \tag{2.10}$$

Similar to the proof of Cheeger inequality, the proof of (2.10) is constructive and leads to a polynomial time algorithm to partition the graph in k subsets whose conductance is at most $O(k^3)\sqrt{\lambda_k}$.

2.5 Additional notation

In this section we introduce additional notation that will be used throughout this thesis. We write $g(n) = \widetilde{O}(f(n))$ as a shorthand for $g(n) = O(f(n)\log^c n)$ for some large enough constant $c > 0$. With a slight abuse of notation, we write $g(n) = \Omega(f(n))$ to indicate that there exists n' such that $g(n) > C \cdot f(n)$ holds for some small enough constant $C > 0$ and for all $n \geq n'$. Analogously, we write $g(n) = \widetilde{\Omega}(f(n))$ to indicate that there exists n' such that $g(n) > C \cdot f(n)\log^c n$ holds for some small enough constants $C, c > 0$ and for all $n \geq n'$.

Finally, we say that an algorithm with an input of size n works in nearly-linear time if the runtime of the algorithm is $\widetilde{O}(n)$. For example, an algorithm that receives as input a graph G of n vertices and m edges works in nearly-linear time if its runtime is $\widetilde{O}(m)$.

Chapter 3

Well-clustered graphs

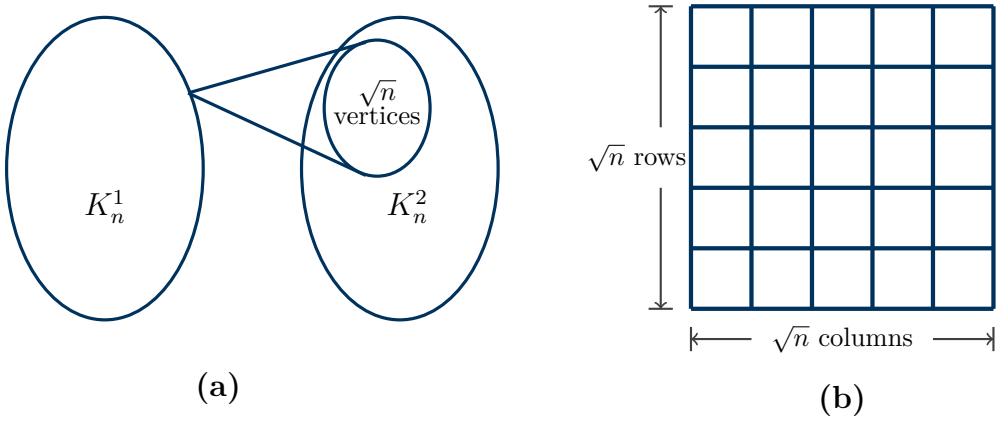
In this chapter we study structural properties of graphs that guarantee the presence of a meaningful cluster-structure. The chapter is organised as follows: in Section 3.1 we formally define well-clustered graphs and discuss the motivation behind this definition. In Section 3.2 we give structural results about the eigenvectors of the Laplacian matrix of well-clustered graphs. These results will be crucial for the design and analysis of the algorithms presented in this thesis. Finally, in Section 3.3 we compare well-clustered graphs with other similar notions proposed in literature, discussing strengths and weaknesses of our definition.

3.1 Formulating a notion of well-clusteredness

In Chapter 2 we introduced the notion of conductance to capture the sparsity of a cut. Informally, we say a set S of vertices forms a cluster if vertices in S are loosely connected to the remaining part of the graph, i.e., if the conductance of S is low. The presence of a subset of low conductance, however, is not sufficient for a graph to possess a meaningful cluster-structure. For instance, let us look at the two graphs depicted in Figure 3.1. Graph G_1 in Figure 3.1(a) consists of two complete graphs K_n^1 and K_n^2 each of n vertices, and every vertex in K_n^1 is connected to $\sqrt{n}$ vertices in K_n^2. For simplicity, we assume $\sqrt{n}$ is integer, and $G_1 = (V, E)$ is defined formally as follows.

- $V = \{(i, j) \colon i = 1, 2; 1 \leq j \leq n\}$
- $E = \Big\{\{(i, j), (i, \ell)\} \colon i = 1, 2; 1 \leq j < \ell \leq n\Big\}$
  $\cup \Big\{\{(1, j), (2, \ell)\} \colon 1 \leq j \leq n; 1 \leq k \leq \sqrt{n} - 1; \ell \equiv j + k \mod n\Big\}.$

In Figure 3.1(b) we have graph G_2, which is simply a square grid with $\sqrt{n}$ rows and $\sqrt{n}$ columns. Notice that, although both graphs have $\Theta(n)$ vertices and the conductance of each graph is $\Theta(1/\sqrt{n})$, which we will prove shortly, G_1 consists of two well-defined clusters, while the cluster-structure of G_2 is not obvious.



(a) **(b)**

Figure 3.1: Graph G_1 consists of two complete graphs of n vertices, in which each vertex from the first complete graph is connected to $\sqrt{n}$ vertices of the second complete graph; graph G_2 is a grid of $\sqrt{n}$ rows and $\sqrt{n}$ columns.

Seeing that the property of having a subset with low conductance alone is not sufficient to provide a well-defined cluster-structure, let us study which condition is further required. To answer this question, notice that, when we decompose G_1 into two parts corresponding to cliques K_n^1 and K_n^2, the subgraph induced from each part has high conductance, while in G_2 we can further split the set with conductance $\Theta(1/\sqrt{n})$ either horizontally or vertically and this gives us two other subsets of the same conductance $\Theta(1/\sqrt{n})$. In other words, a further partition of G_1 into $k \geq 3$ clusters will lead to a dramatic increase of the conductance of a cluster, while this is not the case for G_2. This fact is formally summarised by the lemma below.

Lemma 3.1. *Let G_1, G_2 be the graphs depicted in Figure 3.1(a) and (b). Then, the following holds:*

1. $\rho_{G_1}(2) = O(1/\sqrt{n})$ *and* $\rho_{G_1}(3) = \Theta(1)$*;*
2. $\rho_{G_2}(2) = \Theta(1/\sqrt{n})$ *and* $\rho_{G_2}(3) = O(1/\sqrt{n})$*.*

Proof. For simplicity we assume n is even. For the first statement, let S_1 and S_2 be the two set of vertices corresponding to the two cliques in G_1. Since every vertex has

degree $\Theta(n+\sqrt{n})$, we have that $\mathrm{vol}(S_1) = \Theta(n\cdot(n+\sqrt{n})) = \Theta(n^2)$. Moreover, the number of edges between S_1 and its complement is $w(S_1, V\setminus S_1) = \Theta(n\sqrt{n})$. Therefore,

$$\rho_{G_1}(2) \leq \frac{w(S_1, V\setminus S_1)}{\mathrm{vol}(S_1)} = O\left(1/\sqrt{n}\right).$$

Next we compute $\rho_{G_1}(3)$. Suppose we partition the vertices of G_1 in three nonempty subsets A_1, A_2, A_3. Then there is a set A_i $(1\leq i\leq 3)$ such that $\max\{|A_i\cap S_1|, |A_i\cap S_2|\} \leq n/2$. Without loss of generality, we further assume that $|A_i\cap S_1| \geq |A_i\cap S_2|$. Hence, every vertex in $A_i\cap S_1$ has at least $n/2-1$ neighbours in $A_i\setminus S_1$, which implies $w(A_i, V\setminus A_i) = \Omega(n\cdot|A_i\cap S_1|)$. By our assumption on A_i, it holds that $\phi_{G_1}(A_i) = \Theta(1)$. Since A_1, A_2, A_3 is an arbitrary 3-way partition, we have that $\rho_{G_1}(3) = \Theta(1)$.

For the second statement, let S be the subset of vertices in G_2 that constitute the first $\sqrt{n}/2$ rows of G_2. Since there are $\Theta(n)$ vertices in S and $\Theta(\sqrt{n})$ edges between S and $V\setminus S$, we have that

$$\rho_{G_2}(2) \leq \frac{w(S, V\setminus S)}{\mathrm{vol}(S)} = O(1/\sqrt{n}).$$

It is not difficult to show that S achieves the minimum conductance of the graph, and $\rho_{G_2}(2) = \Theta(1/\sqrt{n})$. Suppose now we divide S in two subsets S_1 and S_2 such that S_1 contains all the vertices of S in the first $\sqrt{n}/2$ columns. A simple computation shows that $S_1, S_2, V\setminus S$ have all $\Theta(n)$ vertices and $\Theta(\sqrt{n})$ outgoing edges. This implies that $\rho_{G_2}(3) = O(1/\sqrt{n})$. □

From the discussion above, we can notice that a small value of $\rho_G(k)$ with a large value of $\rho_G(k+1)$ is sufficient for a graph G to present a cluster-structure. In fact, a large gap between λ_{k+1} and $\rho(k)$ guarantees that G has a cluster-structure, since by the higher-order Cheeger inequality (2.10) a large gap between λ_{k+1} and $\rho(k)$ implies that

1. there exists a k-way partition $\{S_i\}_{i=1}^k$ with bounded conductance $\phi_G(S_i) \leq \rho_G(k)$;
2. any $(k+1)$-way partition of G contains a subset A with conductance $\phi_G(A) \geq \rho_G(k+1) \geq \lambda_{k+1}/2$, which is assumed to be significantly higher than $\rho_G(k)$.

In other words, a large value of $\lambda_{k+1}/\rho_G(k)$ guarantees there exists a large gap between $\rho_G(k)$ and $\rho_G(k+1)$. For this reason, for any connected graph G and $k\geq 2$, we define

the parameter

$$\Upsilon_G(k) \triangleq \frac{\lambda_{k+1}}{\rho_G(k)}, \tag{3.1}$$

and we say that G is *well-clustered* if $\Upsilon_G(k)$ is large for some $k \geq 2$.

To reinforce our intuition on the definition of $\Upsilon_G(k)$, suppose G consists of k connected components. As mentioned in Section 2.3, this implies that $\mathcal{L}_G$ has exactly k eigenvalues equal to zero. Hence, the denominator in (3.1) is zero, and we can assume by convention that $\Upsilon_G(k) = +\infty$. Furthermore, the indicator vectors of the k components correspond to the k bottom eigenvectors of $\mathcal{L}_G$, and so it is straightforward to reconstruct the k components of G using the eigenvectors $f_1, \ldots, f_k$ of $\mathcal{L}_G$. Suppose we now add edges connecting different connected components of G: how many edges can we add before G loses its cluster-structure? Intuitively, it depends not only on the number of edges we add, or where we place them, but also on the connectivity of the k original components. For examples, if G consists of k cliques, more edges need to be added to break the original cluster-structure of G, compared with the case where G consists of k disjoint cycles. Indeed, the numerator in (3.1) informally expresses "how many" edges we added between components, while the denominator expresses the inner-connectivity of the components. Moreover, as long as the subspace spanned by the k indicator vectors of the k components is still close to the subspace spanned by the bottom k eigenvectors of $\mathcal{L}_G$, the cluster-structure of G cannot change that much. We will show in Section 3.2 that this holds inasmuch as $\Upsilon_G(k)$ is large. We also refer to [50] for a more precise discussion of this argument.

3.2 The structure theorem

We now study structural properties of well-clustered graphs. Let $G = (V, E)$ be a graph with a large value of $\Upsilon_G(k)$ for some k, and $\{S_1, \ldots, S_k\}$ be an optimal clustering achieving $\rho_G(k)$. Associated with each cluster, we define an indicator vector $\chi_{S_i} \in \mathbb{R}^n$, where $\chi_{S_i}(v) = 1$ if $v \in S_i$, and $\chi_{S_i}(v) = 0$ otherwise. We assume at the moment that these k clusters are entirely disconnected with each other. In this case, it is easy to see that the first k eigenvectors $f_1, \ldots, f_k$ are the same as $D_G^{1/2}\chi_{S_1}, \ldots, D_G^{1/2}\chi_{S_k}$. Generalising this basic fact, one would expect that, when we add a few edges crossing these k clusters, the set of eigenvectors $\{f_1, \ldots, f_k\}$ and the set of normalised indicator vectors $\{D_G^{1/2}\chi_{S_1}, \ldots, D_G^{1/2}\chi_{S_k}\}$ have similar properties, as long as these k clusters are *loosely* connected. In particular, one can expect that the subspace spanned by

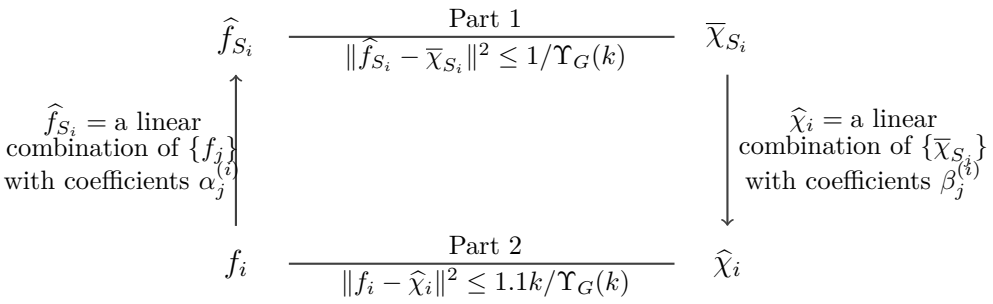


Figure 3.2: Relations among $\{\widehat{f}_{S_i}\}$, $\{f_i\}$, $\{\overline{\chi}_{S_i}\}$, and $\{\widehat{\chi}_i\}$ given by Theorem 3.1.

$f_1, \ldots, f_k$ must be close to the subspace spanned by $D_G^{1/2}\chi_{S_1}, \ldots, D_G^{1/2}\chi_{S_k}$. This fact relates combinatorial properties of well-clustered graphs to their spectral ones, and it is the key for spectral methods to work in the setting of community detection. In this section we will present a rigorous theory behind our intuition above.

Formally, for any $i = 1, \ldots, k$, we define the normalised indicator vector of S_i as

$$\overline{\chi}_{S_i} = \frac{D_G^{1/2}\chi_{S_i}}{\|D_G^{1/2}\chi_{S_i}\|}.$$

Notice that $\|\overline{\chi}_{S_i}\| = 1$ and $\overline{\chi}_{S_i} \perp \overline{\chi}_{S_j}$ for any $i \neq j$. Theorem 3.1 shows that, as long as $\Upsilon_G(k)$ is large, each $\overline{\chi}_{S_i}$ is close to some linear combination $\widehat{f}_{S_i}$ of the eigenvectors $f_1, \ldots, f_k$, and any f_i is close to some linear combination $\widehat{\chi}_i$ of $\overline{\chi}_{S_1}, \ldots, \overline{\chi}_{S_k}$. Equivalently, Theorem 3.1 says that $\operatorname{span}\{\overline{\chi}_{S_1}, \ldots, \overline{\chi}_{S_k}\} \approx \operatorname{span}\{f_1, \ldots, f_k\}$. The relations among the four sets of vectors and their approximation described by Theorem 3.1 are illustrated in Figure 3.2.

Theorem 3.1 (The structure theorem for well-clustered graphs)**.** *Suppose that $\Upsilon_G(k) > C \cdot k$ for a large enough constant $C > 0$. Let $S_1, \ldots, S_k$ be an optimal clustering achieving $\rho_G(k)$. Then, the following statements hold for any $1 \leq i \leq k$:*

1. *There is a linear combination of the eigenvectors $f_1, \ldots, f_k$ with coefficients $\alpha_j^{(i)}$, $\widehat{f}_{S_i} = \alpha_1^{(i)} f_1 + \cdots + \alpha_k^{(i)} f_k$, such that $\left\|\overline{\chi}_{S_i} - \widehat{f}_{S_i}\right\|^2 \leq 1/\Upsilon_G(k)$.*
2. *There is a linear combination of the vectors $\overline{\chi}_{S_1}, \ldots, \overline{\chi}_{S_k}$ with coefficients $\beta_j^{(i)}$, $\widehat{\chi}_i = \beta_1^{(i)}\overline{\chi}_{S_1} + \cdots + \beta_k^{(i)}\overline{\chi}_{S_k}$, such that $\|f_i - \widehat{\chi}_i\|^2 \leq 1.1k/\Upsilon_G(k)$.*

Part 1 of Theorem 3.1 shows that the normalised indicator vector of any cluster S_i is close to some linear combination of the bottom k eigenvectors of $\mathcal{L}_G$. The proof follows from the fact that, since $\overline{\chi}_{S_i}$ has small Rayleigh quotient, it cannot have a large

inner product with eigenvectors corresponding to large eigenvalues. Hence, since the eigenvectors of $\mathcal{L}_G$ span the whole space, $\overline{\chi}_{S_i}$ must have a large inner product with some linear combinations of $f_1, \ldots, f_k$. This statement was also implicitly showed in Theorem 2.2 of [7]. We also remark that Theorem 3.1 can be applied to an arbitrary k-way partition achieving $\rho_G(k)$. As such partitioning may not be unique, this theorem implies that, for all optimal partitions, every cluster has a large overlap with its "correspondence" in another optimal partition, i.e., the clusters are *stable*. Notice that this fact does not hold without a proper assumption on $\Upsilon_G(k)$, and this can be seen as another reason why the parameter $\Upsilon_G(k)$ gives a reasonable characterisation of well-clustered graphs.

Proof of Part 1 of Theorem 3.1. Fix any $1 \leq i \leq k$. By the spectral theorem (see Section 2.2), we can write $\overline{\chi}_{S_i}$ as a linear combination of the eigenvectors of $\mathcal{L}_G$, i.e.,

$$\overline{\chi}_{S_i} = \alpha_1^{(i)} f_1 + \cdots + \alpha_n^{(i)} f_n$$

for some scalars $\alpha_1^{(i)}, \ldots, \alpha_n^{(i)} \in \mathbb{R}$. Let $\widehat{f}_{S_i}$ be the projection of $\overline{\chi}_{S_i}$ on the subspace spanned by $\{f_i\}_{i=1}^k$, i.e.,

$$\widehat{f}_{S_i} = \alpha_1^{(i)} f_1 + \cdots + \alpha_k^{(i)} f_k.$$

We want to show that $\widehat{f}_{S_i}$ is close to $\overline{\chi}_{S_i}$. By the definition of Rayleigh quotients (2.3), we have that

$$\begin{aligned}
\mathcal{R}_G(\chi_{S_i}) &= \overline{\chi}_{S_i}^{\mathsf{T}} \mathcal{L}_G \overline{\chi}_{S_i} \\
&= \left(\alpha_1^{(i)} f_1 + \cdots + \alpha_n^{(i)} f_n\right)^{\mathsf{T}} \mathcal{L}_G \left(\alpha_1^{(i)} f_1 + \cdots + \alpha_n^{(i)} f_n\right) \\
&= \left(\alpha_1^{(i)}\right)^2 \lambda_1 + \cdots + \left(\alpha_n^{(i)}\right)^2 \lambda_n \\
&\geq \left(\alpha_1^{(i)}\right)^2 \lambda_1 + \cdots + \left(\alpha_k^{(i)}\right)^2 \lambda_k + (1 - \alpha') \lambda_{k+1} \\
&\geq (1 - \alpha') \lambda_{k+1}, \qquad (3.2)
\end{aligned}$$

where we used the orthonormality of $f_1, \ldots, f_n$ and

$$\alpha' \triangleq \left(\alpha_1^{(i)}\right)^2 + \cdots + \left(\alpha_k^{(i)}\right)^2.$$

By (2.8) and the optimality of $S_1, \ldots, S_k$, $\mathcal{R}_G(\chi_{S_i})$ satisfies

$$\mathcal{R}_G(\chi_{S_i}) \leq \rho_G(k).$$

Therefore, by (3.2) and the definition of $\Upsilon_G(k)$,

$$1 - \alpha' \leq \frac{\mathcal{R}_G(\chi_{S_i})}{\lambda_{k+1}} \leq \frac{\rho_G(k)}{\lambda_{k+1}} = \frac{1}{\Upsilon_G(k)} \tag{3.3}$$

and

$$\left\|\overline{\chi}_{S_i} - \widehat{f}_{S_i}\right\|^2 = \left(\alpha_{k+1}^{(i)}\right)^2 + \cdots + \left(\alpha_n^{(i)}\right)^2 = 1 - \alpha' \leq 1/\Upsilon_G(k),$$

which finishes the proof. □

Part 2 of Theorem 3.1 is more interesting, and shows the opposite direction holds as well, i.e., any f_i $(1 \leq i \leq k)$ can be approximated by a linear combination of the normalised indicator vectors $\{\overline{\chi}_{S_i}\}_{i=1}^k$. To sketch the proof, note that if we could write every $\overline{\chi}_{S_i}$ *exactly* as a linear combination of $\{f_i\}_{i=1}^k$, then we could write every f_i $(1 \leq i \leq k)$ as a linear combination of $\{\overline{\chi}_{S_i}\}_{i=1}^k$. This is because it would hold that $\operatorname{span}\left\{\overline{\chi}_{S_1}, \ldots, \overline{\chi}_{S_k}\right\} \subseteq \operatorname{span}\{f_1, \ldots, f_k\}$, and both of $\{f_i\}_{i=1}^k$ and $\{\overline{\chi}_{S_i}\}_{i=1}^k$ are sets of linearly independent vectors of the same dimension. Part 1, however, only shows that the $\overline{\chi}_{S_i}$'s are close to a linear combination $\widehat{f}_{S_i}$ of the first k eigenvectors. Nevertheless, this is enough to show that these $\{\widehat{f}_{S_i}\}_{i=1}^k$ are almost orthogonal between each other, and, as a consequence, that $\operatorname{span}\{\widehat{f}_{S_1}, \ldots, \widehat{f}_{S_k}\} = \operatorname{span}\{f_1, \ldots, f_k\}$, which will imply Part 2.

We will use the following classical result in our proof.

Theorem 3.2 (Geršgorin Circle Theorem [29])**.** *Let A be an $n \times n$ matrix, and let $R_i(A) = \sum_{j \neq i} |A_{i,j}|$, for $1 \leq i \leq n$. Then, all eigenvalues of A are in the union of Geršgorin Discs defined by*

$$\bigcup_{i=1}^{n} \{z \in \mathbb{C} : |z - A_{i,i}| \leq R_i(A)\}.$$

Proof of Part 2 of Theorem 3.1. By Part 1, every $\overline{\chi}_{S_i}$ is approximated by a vector $\widehat{f}_{S_i}$ defined by

$$\widehat{f}_{S_i} = \alpha_1^{(i)} f_1 + \cdots + \alpha_k^{(i)} f_k.$$

We start showing that the vectors $\{\widehat{f}_{S_j}\}_{j=1}^k$ are linearly independent. To this aim, we define a $k \times k$ matrix A such that $A_{i,j} = \alpha_i^{(j)}$, i.e., the jth column of matrix A consists of values $\left\{\alpha_i^{(j)}\right\}_{i=1}^k$ representing $\widehat{f}_{S_j}$. We express the jth column of A by a vector $\alpha^{(j)}$, defined as

$$\alpha^{(j)} = \left(\alpha_1^{(j)}, \cdots, \alpha_k^{(j)}\right)^\intercal.$$

We will show that the vectors $\left\{\alpha^{(j)}\right\}_{j=1}^{k}$ are linearly independent, which implies that $\{\widehat{f}_{S_j}\}_{j=1}^{k}$ are linearly independent as well. To prove this, we will show that $A^\intercal A$ has no zero eigenvalue, and hence A is invertible and its columns are linearly independent.

We want to bound the inner products $\left\langle \alpha^{(i)}, \alpha^{(j)} \right\rangle$ for $i \neq j$. We start showing that $\left|\left\langle \overline{\chi}_{S_i} - \widehat{f}_{S_i}, \overline{\chi}_{S_j} \right\rangle\right|$ is small. By definition we have that

$$\begin{aligned}\left|\left\langle \overline{\chi}_{S_i} - \widehat{f}_{S_i}, \overline{\chi}_{S_j} \right\rangle\right| &= \left|\left\langle \alpha^{(i)}_{k+1} f_{k+1} + \cdots + \alpha^{(i)}_{n} f_n, \alpha^{(j)}_{1} f_1 + \cdots + \alpha^{(j)}_{n} f_n \right\rangle\right| \\ &= \alpha^{(i)}_{k+1}\alpha^{(j)}_{k+1} + \cdots + \alpha^{(i)}_{n}\alpha^{(j)}_{n} \end{aligned} \tag{3.4}$$

where the last equality follows from the orthonormality of $f_1, \ldots, f_n$. Since it holds for any $1 \leq \ell \leq k$ that $\left(\alpha^{(\ell)}_{k+1}\right)^2 + \cdots + \left(\alpha^{(\ell)}_{n}\right)^2 \leq 1/\Upsilon_G(k)$, (3.4) implies that

$$\left|\left\langle \overline{\chi}_{S_i} - \widehat{f}_{S_i}, \overline{\chi}_{S_j} \right\rangle\right| \leq 1/\Upsilon_G(k). \tag{3.5}$$

We can now turn to study $\left|\left\langle \alpha^{(i)}, \alpha^{(j)} \right\rangle\right|$. First of all, notice that it holds by the orthonormality of $\{f_i\}_{i=1}^{k}$ that

$$\begin{aligned}&\left|\left\langle \alpha^{(i)}, \alpha^{(j)} \right\rangle\right| \\ &= \left|\left\langle \widehat{f}_{S_i}, \widehat{f}_{S_j} \right\rangle\right| = \left|\left\langle \overline{\chi}_{S_i} - (\overline{\chi}_{S_i} - \widehat{f}_{S_i}), \overline{\chi}_{S_j} - (\overline{\chi}_{S_j} - \widehat{f}_{S_j}) \right\rangle\right| \\ &= \left|\left\langle \overline{\chi}_{S_i}, \overline{\chi}_{S_j} \right\rangle - \left\langle \overline{\chi}_{S_i} - \widehat{f}_{S_i}, \overline{\chi}_{S_j} \right\rangle - \left\langle \overline{\chi}_{S_i}, \overline{\chi}_{S_j} - \widehat{f}_{S_j} \right\rangle + \left\langle \overline{\chi}_{S_i} - \widehat{f}_{S_i}, \overline{\chi}_{S_j} - \widehat{f}_{S_j} \right\rangle\right| \\ &\leq 3/\Upsilon_G(k), \end{aligned} \tag{3.6}$$

where we used in the last inequality the orthogonality of $\overline{\chi}_{S_i}$ and $\overline{\chi}_{S_j}$, (3.4), and (3.5). So it holds for any $i \neq j$ that

$$|(A^\intercal A)_{i,j}| = \left|\sum_{\ell=1}^{k} A_{\ell,i} A_{\ell,j}\right| = \left|\sum_{\ell=1}^{k} \alpha^{(i)}_{\ell}\alpha^{(j)}_{\ell}\right| = \left|\left\langle \alpha^{(i)}, \alpha^{(j)} \right\rangle\right| \leq 3/\Upsilon_G(k), \text{ and}$$

$$(A^\intercal A)_{i,i} = \sum_{\ell=1}^{k} \left(\alpha^{(i)}_{\ell}\right)^2 = \left\|\overline{\chi}_{S_i}\right\|^2 - \left\|\overline{\chi}_{S_i} - \widehat{f}_{S_i}\right\|^2 \geq 1 - 1/\Upsilon_G(k).$$

Then, by Theorem 3.2 it holds that all the eigenvalues of $A^\intercal A$ are at least $1 - 1/\Upsilon_G(k) - 3 \cdot (k-1)/\Upsilon_G(k)$. Therefore, A has no eigenvalue with value 0 as long as $\Upsilon_G(k) > 10k$. With this we proved that the vectors $\left\{\alpha^{(j)}\right\}_{j=1}^{k}$. Hence, the vectors $\{\widehat{f}_{S_i}\}_{i=1}^{k}$ are linearly independent as well, since $\left\langle \alpha^{(i)}, \alpha^{(j)} \right\rangle = \left\langle \widehat{f}_{S_i}, \widehat{f}_{S_j} \right\rangle$.

We now combine this with the fact that $\operatorname{span}\{\widehat{f}_{S_1},\dots,\widehat{f}_{S_k}\} \subseteq \operatorname{span}\{f_1,\dots,f_k\}$ and $\dim(\operatorname{span}(\{\widehat{f}_{S_1},\dots,\widehat{f}_{S_k}\})) = k$, showing that

$$\operatorname{span}\{\widehat{f}_{S_1},\dots,\widehat{f}_{S_k}\} = \operatorname{span}\{f_1,\dots,f_k\}.$$

Hence, we can write every f_i $(1 \le i \le k)$ as a linear combination of $\{\widehat{f}_{S_i}\}_{i=1}^k$, i.e.,

$$f_i = \beta_1^{(i)}\widehat{f}_{S_1} + \cdots + \beta_k^{(i)}\widehat{f}_{S_k}. \tag{3.7}$$

The purpose of this proof is to construct vectors $\widehat{\chi}_i \in \operatorname{span}\{\overline{\chi}_{S_1},\dots,\overline{\chi}_{S_k}\}$ for $1 \le i \le k$ that are close to f_i. For this reason, we take (3.7) and we simply substitute any $\widehat{f}_{S_i}$ with its close approximation $\overline{\chi}_{S_i}$:

$$\widehat{\chi}_i = \beta_1^{(i)}\overline{\chi}_{S_1} + \cdots + \beta_k^{(i)}\overline{\chi}_{S_k}. \tag{3.8}$$

It remains to show that $\|f_i - \widehat{\chi}_i\|$ is small. We first need to bound the norm of $\beta^{(i)} \triangleq \left(\beta_1^{(i)},\dots,\beta_k^{(i)}\right)^\intercal$. Notice that

$$\begin{aligned}
1 = \|f_i\|^2 &= \sum_{\ell=1}^k \left(\beta_\ell^{(i)}\right)^2 \left\|\widehat{f}_{S_\ell}\right\|^2 + \sum_{\ell\neq\ell'} \beta_\ell^{(i)}\beta_{\ell'}^{(i)} \left\langle \widehat{f}_{S_\ell}, \widehat{f}_{S_{\ell'}} \right\rangle \\
&\ge \|\beta^{(i)}\|^2(1 - 1/\Upsilon_G(k)) - \sum_{\ell} \left|\beta_\ell^{(i)}\right| \sum_{\ell'\neq\ell} \left|\beta_{\ell'}^{(i)}\right| \left\langle \widehat{f}_{S_\ell}, \widehat{f}_{S_{\ell'}} \right\rangle \\
&\ge \|\beta^{(i)}\|^2(1 - 1/\Upsilon_G(k)) - \left(\sqrt{k}\cdot\|\beta^{(i)}\|\right)\cdot\left(\sqrt{k}\cdot\|\beta^{(i)}\|\right)\cdot 3/\Upsilon_G(k) \\
&\ge (1 - 1/\Upsilon_G(k) - 3k/\Upsilon_G(k))\,\|\beta^{(i)}\|^2,
\end{aligned}$$

where the first inequality holds since, by orthogonality, $\left\|\overline{\chi}_{S_\ell}\right\|^2 = \left\|\widehat{f}_{S_\ell}\right\|^2 + \left\|\widehat{f}_{S_\ell} - \overline{\chi}_{S_\ell}\right\|^2$, while the second holds by the Cauchy-Schwarz inequality and (3.6). Since $\Upsilon_G(k) > C\cdot k$ for some big enough $C > 0$, we have that

$$\|\beta^{(i)}\|^2 \le \left(1 - \frac{1}{\Upsilon_G(k)} - \frac{3k}{\Upsilon_G(k)}\right)^{-1} \le 1 + \frac{4k}{\Upsilon_G(k)} \le 1.1. \tag{3.9}$$

Combining this with Part 1 of Theorem 3.1 and the Cauchy-Schwarz inequality,

$$\|f_i - \widehat{\chi}_i\| \le \sum_{j=1}^k \left|\beta_j^{(i)}\right| \left\|\widehat{f}_{S_j} - \overline{\chi}_{S_j}\right\| \le \left(1/\sqrt{\Upsilon_G(k)}\right) \sum_{j=1}^k \left|\beta_j^{(i)}\right| \le \sqrt{1.1k/\Upsilon_G(k)},$$

which, after squaring both sides of the inequality, proves Part 2 of the theorem. □

3.3 Further discussions

In the previous two sections we introduced the notion of well-clustered graphs and proved Theorem 3.1 which relates the eigenvectors of the Laplacian of well-clustered graphs to their cluster-structure. In this section we further discuss these notions: we first compare our definition of well-clustered graphs to other assumptions studied in literature. Then we discuss the differences between Theorem 3.1 and standard matrix perturbation theorems. We end the chapter with a discussion of stochastic block models.

3.3.1 Comparison between different clusterability assumptions

We argued that a reasonable gap between λ_{k+1} and $\rho(k)$ implies that a well-defined structure of clusters of a graph. Now we compare our definition of $\Upsilon_G(k)$ with similar assumptions proposed in literature.

Clusterability assumption based on λ_{k+1} vs λ_k. A significant gap between two consecutive eigenvalues λ_k and λ_{k+1} has been widely used in practice to determine the number of clusters in a graph [74]. A formal proof of this fact, however, is missing (see, e.g., [53]). Indeed, by the higher-order Cheeger inequality a quadratic gap between λ_k and λ_{k+1} implies our assumption on $\Upsilon_G(k)$, but it remains an important open question to see whether a constant multiplicative gap between two consecutive eigenvalues implies the graph has a meaningful cluster-structure.

Clusterability assumption based on $\rho(k+1)$ vs $\rho(k)$. We mentioned that if a graph can be partitioned in k clusters of low conductance, but every $(k+1)$-way partition has at least a cluster with significantly higher conductance, i.e., $\rho(k+1) \gg \rho(k)$, then we can assume the graph has a meaningful cluster-structure. This condition has been studied in [53], which shows that a large enough gap between $\rho_G(k)$ and $\rho_G(k+1)$ implies the existence of a partition $\{S_i\}_{i=1}^k$ of the vertices of G such that, for any S_i in the partition, there exists a gap between the *outer*-conductance $\phi_G(S_i)$ and the *inner*-conductance $\phi_{G[S_i]}$, where $\phi_{G[S_i]}$ is the conductance of the subgraph of G induced by the vertices in S_i. This result, however, is just existential, and to recover the clusters with provable guarantees, the algorithm in [53] needs to assume at least a quadratic gap between the eigenvalues of $\mathcal{L}_G$, i.e., $\lambda_{k+1} = \Omega(\sqrt{\lambda_k})$, which, by the higher-order Cheeger inequality (2.10), already implies a large value of $\Upsilon_G(k)$. In fact,

this quadratic dependence is necessary for many spectral algorithms to work. Indeed, we can show the following fact:

> *There are graphs with $\rho_G(k+1) \gg \rho_G(k)$, which implies by [53] a well-defined cluster-structure, but for which $\Upsilon_G(k)$ is small and classical spectral approaches, including Algorithm 1, are not able to return a good clustering.*

For instance, we look at the graph $\mathsf{Grid} = (V, E, w)$ in Figure 3.3. It consists of $\sqrt{n}$ rows, and $3\sqrt{n}$ columns, where $\sqrt{n}$ is an even number, and the sets V and E are defined as follows:

$$V = \left\{(i,j) | 1 \leq i \leq \sqrt{n}, 1 \leq j \leq 3\sqrt{n}\right\}$$
$$E = \left\{((i,j),(i',j')) \mid i = i' \wedge (j - j' \equiv 1 \bmod 3\sqrt{n}), \text{or } j = j' \wedge (i - i' \equiv 1 \bmod \sqrt{n})\right\}.$$

The weight of every edge $((i,j),(i',j'))$ equals to $w((i,j),(i',j')) = 1$, except for the edges sitting in the middle row, for which we set $w((i,j),(i',j')) = 1/\sqrt{n}$ if $i = \sqrt{n}/2$ and $i' = \sqrt{n}/2 + 1$. While this graph does present two well-defined clusters, which divide the graph halfway horizontally, and there is a large gap between $\rho_{\mathsf{Grid}}(2)$ and $\rho_{\mathsf{Grid}}(3)$, $\Upsilon_{\mathsf{Grid}}(2)$ is small.

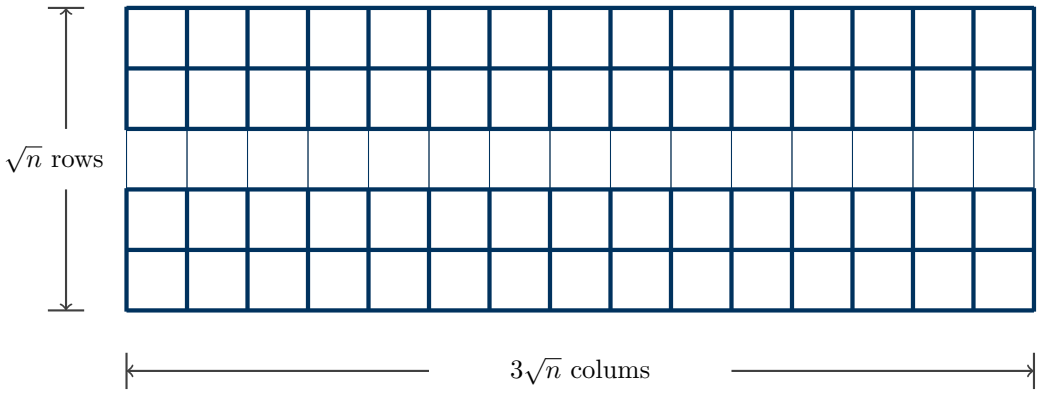


Figure 3.3: Graph Grid: A $\sqrt{n} \times 3\sqrt{n}$ grid in which (i) the weight of the edges between the two middle rows is $1/\sqrt{n}$, and (ii) the weight of all the other edges is 1. An interesting fact about this graph is that, while the sparsest cut is the one dividing the graph halfway *horizontally*, the cut returned by Algorithm 1 divides the grid halfway *vertically*.

Fact 3.1. *It holds that $\rho_{\mathsf{Grid}}(2) = \Theta(1/n)$, $\rho_{\mathsf{Grid}}(3) = \Omega(1/\sqrt{n})$, and $\Upsilon_{\mathsf{Grid}}(2) = O(1)$.*

Proof. We define a set S by

$$S = \{(i,j) | (i,j) \in V[\mathsf{Grid}] \wedge (i \leq \sqrt{n}/2)\}.$$

By construction, it holds that $w(S, V \setminus S) = \Theta(1)$ and $\mathrm{vol}(S) = \Theta(n)$. Therefore,

$$\rho_{\mathsf{Grid}}(2) \leq \frac{w(S, V \setminus S)}{\mathrm{vol}(S)} = O\left(\frac{1}{n}\right).$$

It is not difficult to prove that S achieves $\rho_{\mathsf{Grid}}(2)$, i.e., $\rho_{\mathsf{Grid}}(2) = \Theta(1/n)$.

Let's now consider any 3-way partitioning S_1, S_2, S_3 of the vertex set of Grid. Then, there exists a set S_i in the partition containing at most $3n/4$ vertices in the upper half of the grid and at most $3n/4$ vertices in the bottom half of the grid, i.e., $|S_i \cap S| \leq 3n/4$ and $|S_i \cap (V \setminus S)| \leq 3n/4$. Clearly, there must be $\Omega\left(\sqrt{|S_i|}\right)$ edges of weight 1 between S_i and $V \setminus S_i$. Therefore, $\rho_{\mathsf{Grid}}(3) \geq w(S_i, \mathrm{vol}(S_i))/\mathrm{vol}(S_i) = \Omega\left(\sqrt{|S_i|}/|S_i|\right) = \Omega(1/\sqrt{n})$.

To bound $\lambda_3(\mathsf{Grid})$, notice that the third smallest eigenvalue of a $\sqrt{n} \times 3\sqrt{n}$ grid in which every edge has weight 1 is of the order $O(1/n)$ [34], and Grid can be obtained from such a graph by decreasing the weight of some of its edges without significantly changing the degree of the vertices. Therefore, we have $\lambda_3(\mathsf{Grid}) = O(1/n)$ (this can be seen by (2.4)). This implies $\Upsilon_{\mathsf{Grid}}(2) = \lambda_3/\rho_{\mathsf{Grid}}(2) = O(1)$. □

From Fact 3.1 we know that $\rho_{\mathsf{Grid}}(3) \gg \rho_{\mathsf{Grid}}(2)$, which implies that Grid has a well-defined cluster-structure, however for this graph $\Upsilon_{\mathsf{Grid}}(3) = O(1)$, i.e., it is not well-clustered in our sense. But this is not necessarily a weakness in our definition of well-clustered graphs. We are interested in providing a *sufficient* condition so that simple spectral methods would be able to recover the cluster-structure of a graph, and Grid doesn't satisfy these conditions: it can be shown Algorithm 1 will split Grid halfway *vertically*, producing a partition with conductance $\Theta(1/\sqrt{n})$, while the optimal partition is the one cutting Grid halfway *horizontally*, which we proved has conductance $\Theta(1/n)$. Compare it to the graph G_1 in Figure 3.1(a), which has essentially the same ratio between $\rho(3)$ and $\rho(2)$, but a larger $\Upsilon_{G_1}(2) = \Theta(\sqrt{n})$: spectral partitioning outputs the optimal partition in G_1, but performs badly for Grid. More examples and exhaustive discussion of graphs for which spectral methods fail to recover a good partitioning can be found in [27].

Other clusterability assumptions. Kannan et al. [32] proposed a bi-criteria optimisation algorithm which aims to find k clusters with high inner-conductance while

minimising the number of edges between different clusters. As pointed out by Oveis Gharan and Trevisan [53], this criteria does not always capture the *best* partitioning of the graph, see [53] for further details.

Another assumption conceptually similar to ours was proposed by Allen Zhu et al. [5] in the context of local graph partitioning: they present an algorithm based on PageRank that is able to obtain a very good approximation of a cluster as long as there is a gap between the outer-conductance of the cluster and its *local mixing time*, which is the time needed until a random walk that starts in a cluster is close to be stationary, conditioned on the fact that it never leaves the cluster in the first place. This assumption can be considered as a local variant of our assumption on $\Upsilon_G(k)$, in the sense that it is related to the inner- and outer-connectivity of a *single* cluster.

Clusterability of geometric datasets. Clusterability assumptions were actually first introduced in the context of k-means clustering. For example, Ostrovsky et al. [52] show that a sharp drop of the k-means objective function when the data is partitioned in $k-1$ or in k clusters implies one can recover a very good approximation of the optimal k-means clustering using simple Lloyd-style heuristics [41]. This can be seen as a geometric equivalent of the assumption based on $\rho(k)$ and $\rho(k+1)$ discussed above. However, for a k-means dataset to be well-clustered, we need a large change in the objective function when we partition the points in k or $k-1$ clusters, while in graphs we require a large change in the objective function when we partition a graph in k or $k+1$ clusters. This is a consequence of the fact that the k-means cost decreases when we increase the value of k, while in graphs $\rho(k)$ is an increasing function of k.

Limitations of our clusterability assumption. We remark that spectral algorithms for graph clustering indeed work for certain classes of graphs that are not captured by our assumption on $\Upsilon_G(k)$. In particular, graphs generated from stochastic block models [28] possess strong regularities which make them easy to cluster even when $\Upsilon_G(k)$ is fairly small. A more detailed comparison between well-clustered graphs and stochastic block models will be presented in Section 3.3.3.

3.3.2 Theorem 3.1 vs the Davis-Kahan theorem

We now compare our structure theorem with the Davis-Kahan theorem [19], which is a fundamental result in matrix perturbation theory, and studies how spectral properties

of a matrix change after being perturbed by another matrix of small spectral norm. The formal statement of the Davis-Kahan theorem is as follows:

Theorem 3.3 (Davis-Kahan $\sin\theta$ theorem, [19])**.** *Let A, B be symmetric $n \times n$ matrices and $E = B - A$. Let $\alpha_1 \leq \cdots \leq \alpha_n$ be the eigenvalues of A with corresponding orthonormal eigenvectors $x_1, \ldots, x_n$, and let $\beta_1 \leq \cdots \leq \beta_n$ be the eigenvalues of B with corresponding orthonormal eigenvectors $y_1, \ldots, y_n$. Let θ_i be the angle between x_i and y_i. Then, for $1 \leq i \leq n$,*

$$\sin 2\theta_i \leq \frac{\|E\|}{\min_{j \neq i} |\beta_i - \beta_j|}.$$

Theorem 3.3 states that the ith eigenvector x_i of the original matrix A is close to the ith eigenvector y_i of the perturbed matrix B, as long as $\|E\|$ is significantly smaller than the *spectral gap* $\min_{j \neq i} |\beta_j - \beta_i|$. This theorem is naturally related to graph clustering, in the sense that one can naturally encode the information about edge connections within the clusters into matrix A, while the information about the edge connections among different clusters into matrix E. Hence, it is logical to ask to which extent our structure theorem gives a better bound than the Davis-Kahan theorem.

To address this question, we first notice the following distinction on the approximation guarantee given by the Davis-Kahan theorem and our structure theorem:

- In the Davis-Kahan theorem the distance between the eigenvectors of the original and the new perturbed matrix is bounded with respect to $\|E\|$.
- When we view the Laplacian of a well-clustered graph G with k clusters as the Laplacian of a graph with k disjoint components perturbed by some matrix representing crossing edges, Theorem 3.1 bounds the distance between the eigenvectors of the Laplacian with k disjoint components and the eigenvectors of $\mathcal{L}_G$ with respect to $\rho_G(k)$.

As shown from the following example, this difference could become crucial to analyse spectral partitioning for certain graphs: let graph G be the union of two regular graphs H_1 and H_2 of size $\Theta(n)$ and degree $d = \Theta(\sqrt{n})$, such that $\rho_{H_1} = \rho_{H_2} = \Theta(1/\sqrt{n})$, $\lambda_2(\mathcal{L}_{H_1}) = \lambda_2(\mathcal{L}_{H_2}) = \Theta(1/\sqrt{n})$, and $\lambda_3(\mathcal{L}_{H_1}) = \lambda_3(\mathcal{L}_{H_2}) = \Omega(1)$.[1] We further assume H_1 and H_2 are connected by $o(\sqrt{n})$ edges in G. This implies that $\rho_G(2) = o(1/n)$ and $\lambda_3(\mathcal{L}_G) = \Omega(1/\sqrt{n})$. Therefore, $\Upsilon_G(3) = \Omega(\sqrt{n})$, and Theorem 3.1 shows that the

[1] it is not difficult to construct such graphs.

second eigenvector of G is close to $\chi_{H_1} - \chi_{H_2}$, where χ_{H_i} is the indicator vector of the vertex set of H_i. However, we will show that Theorem 3.3 gives us little information when we analyse the cluster-structure of G.

To apply Theorem 3.3, we first notice that G is *almost*-regular and for convenience, instead of $\mathcal{L}_G$, we work with the matrix

$$\mathbb{I} - \frac{1}{\sqrt{n}} \cdot A_G,$$

where A_G is the adjacency matrix of G. We further decompose this matrix as

$$\mathbb{I} - \frac{1}{\sqrt{n}} A_G = \mathbb{I} - \frac{1}{\sqrt{n}} \left(A_{H_1} + A_{H_2} + E \right),$$

where A_{H_1}, A_{H_2} are the $n \times n$ matrices that encodes the edge connections of H_1 and H_2, and E encodes the information about the edges between H_1 and H_2. Since

$$|\lambda_2(\mathcal{L}_G) - \lambda_1(\mathcal{L}_G)| = O(1/\sqrt{n}),$$

Theorem 3.3 presents a non-trivial result about the eigenvectors of $\mathbb{I} - \frac{1}{\sqrt{n}} A_G \approx \mathcal{L}_G$ only if

$$\frac{\|E\|}{\sqrt{n} \cdot |\lambda_2(\mathcal{L}_G) - \lambda_1(\mathcal{L}_G)|} = \Omega(\|E\|)$$

is small. But this means that even adding a constant number of edges between H_1 and H_2 could make Theorem 3.3 meaningless!

Generalising the above example, it is easy to see that the bound given by Theorem 3.3 is not sufficient when the perturbation matrix has high spectral norm comparing with the spectral gap. However, the structure theorem can be successfully applied in this setting, since the norm of the bottom k eigenvectors of a graph Laplacian is approximately distributed uniformly over all clusters, rather than being localised on a small subset of vertices. In other words, even if we apply a perturbation matrix with high norm but this perturbation does not change the cluster-structure of the graph, the bottom eigenspace will not be influenced significantly. One novelty of Theorem 3.1 is that it takes into account the particular structure of the bottom eigenvectors of the graph Laplacian, and hence outperforms standard matrix perturbation theorems for many applications.

3.3.3 Stochastic block models

At the end of this chapter, we highlight that a large value of $\Upsilon_G(k)$ is only a sufficient condition for spectral methods to work well. In particular, there are several classes of graphs for which spectral methods work well, although these graphs have a small value of $\Upsilon_G(k)$. These are usually graphs with strong regularities, e.g., graphs generated by random models. A typical example are Stochastic Block Models (SBMs) [28], which are generative models of random graphs with a community-structure. The basic model consists of two unknown communities X, Y of the same size, and parameters p, q satisfying $0 < q < p < 1$. These two parameters represent the probabilities that there is an edge between two vertices from the same community, and from different communities. Formally, for any pair of vertices (u, v), we place an edge between u and v with the following probability distribution:

$$\mathbb{P}[\text{there is an edge between } u \text{ and } v] = \begin{cases} p & \text{if } u, v \in X \\ p & \text{if } u, v \in Y \\ q & \text{if } u \in X, v \in Y. \end{cases}$$

We write $G \sim \mathcal{G}_n(p, q)$ to indicate that $G = (V, E)$ is sampled from this model and we assume $V = X \cup Y, |X| = |Y| = n/2$.

The stochastic block model is one of the simplest models for community detection and, even though it is not an accurate model for real-world graphs, these graphs form an important benchmark for community-detection algorithms, and have a rich literature behind them (see [1] for recent developments in the area). These references include a study of the conditions on p and q under which spectral and semi-spectral methods work [12, 16, 18, 48], the conditions on p and q for which a reasonable/perfect recovery is possible, the generalisation of this model to $k > 2$ clusters [2, 58], graphs with non-regular degree sequences [57], as well as connections to statistical physics [21, 45, 49].

When p is significantly larger than q and $p > C' \cdot \log n/n$ for a large enough constant C', $G \sim \mathcal{G}_n(p, q)$ is a well-clustered graph with high probability [58]. Spectral techniques, however, have been shown to work even when p is very close to q [40], and G is not well-clustered in our sense. The following argument due to [48] explains the reason behind this scenario.

Let $G \sim \mathcal{G}_n(p, q)$. We observe that the adjacency matrix A_G is close in expectation to a rank-2 matrix. Let $\bar{A}_G = A_G + p\mathbb{I}$. Notice that $\bar{A}_G$ has the same eigenvectors as A_G and its eigenvalues are just the eigenvalues of A_G shifted by p, and the expectation

of A_G can be written as

$$M \triangleq \mathbb{E}\left[\bar{A}_G\right] = \begin{bmatrix} p & \cdots & p & q & \cdots & q \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ p & \cdots & p & q & \cdots & q \\ q & \cdots & q & p & \cdots & p \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ q & \cdots & q & p & \cdots & p \end{bmatrix} \tag{3.10}$$

Let $\mathbf{1}$ be a unit vector constant on all the vertices of G, i.e., $\mathbf{1}(u) = 1/\sqrt{n}$ for any $u \in V$. Clearly, $\mathbf{1}$ is an eigenvector for $\mathbb{E}\left[\bar{A}_G\right]$ with eigenvalues $n(p+q)/2$. Consider now a unit vector $y \in \mathbb{R}^n$ whose entries have the same absolute value but opposite sign on the two communities. More precisely,

$$y(u) = \begin{cases} \frac{1}{\sqrt{n}} & \text{if } u \in X \\ -\frac{1}{\sqrt{n}} & \text{if } u \in Y. \end{cases} \tag{3.11}$$

Since $y \perp \mathbf{1}$, M can be decomposed in the following way:

$$M = n \cdot \frac{p+q}{2} \cdot \mathbf{1}\mathbf{1}^\intercal + n \cdot \frac{p-q}{2} \cdot yy^\intercal.$$

This means M has only two nonzero eigenvalues and, as long as $p > q$, they are distinct. Moreover, the eigenvector corresponding to the smallest nonzero eigenvalue clearly contains all the information to exactly recover the two ground-truth communities. Of course, we don't have access to M, but we can use arguments from random matrix theory to show that A_G, as long as p is not too small, is close in spectral norm to its expectation. Combining this with Theorem 3.3, we can show that, if p is not too close to q compared to the spectral norm of $M - A_G$, the eigenvectors of A_G are close to the eigenvectors of M. As a consequence, we can approximately recover the two communities X, Y from the eigenvectors of A_G.

Hence, as the eigenvectors of $\mathbb{E}[A_G]$ encode all the information about X and Y, we only need to ensure $p > q$ to recover the two clusters, and in theory this holds independent of how close p and q are. However, unless $p \gg q$, the generated graphs are not well-clustered and hence Theorem 3.1 cannot be applied. Hence, it is clear that the SBMs are endowed with regularities that can be exploited by spectral methods but that cannot be captured by Theorem 3.1.

Chapter 4

Approximation guarantees for Spectral Clustering

4.1 Introduction

Spectral Clustering is one of the most popular algorithms for graph partitioning [74]. It was popularised at the beginning of the 2000s [50, 60] and has enjoyed considerable success since then. The general framework consists in (1) computing an embedding of the vertices in a low dimensional Euclidean space using the eigenvectors of a matrix representing the graph, (2) partitioning these points using a geometric clustering algorithm, such as k-means, (3) returning a corresponding partition of the graph.

In this chapter, we consider the following specific version of Spectral Clustering. Given a graph $G = (V, E, w)$ with normalised Laplacian $\mathcal{L}_G$, let $f_1, \ldots, f_k$ be the bottom k eigenvectors of $\mathcal{L}_G$. The *spectral embedding* $F : V \to \mathbb{R}^k$ maps any vertex $u \in V$ to a point

$$F(u) = \frac{1}{\sqrt{d_u}} \left(f_1(u), \ldots, f_k(u)\right)^\intercal . \tag{4.1}$$

Spectral Clustering computes the embedding $\{F(u)\}_{u \in V}$ and partitions these points with a k-means algorithm. Given a set of points $\mathcal{X} \subset \mathbb{R}^d$, a k-means algorithm seeks to find a set $\mathcal{K} \subset \mathbb{R}^d$ of k centres $c_1, \cdots, c_k$ in order to minimise the sum of the ℓ_2^2-distance between $x \in \mathcal{X}$ and the centre to which it is assigned. Formally, for any partition $\mathcal{X}_1, \cdots, \mathcal{X}_k$ of the set $\mathcal{X} \subset \mathbb{R}^d$, we define the cost function as

$$\mathsf{COST}(\mathcal{X}_1, \ldots, \mathcal{X}_k) \triangleq \min_{c_1, \ldots, c_k \in \mathbb{R}^d} \sum_{i=1}^{k} \sum_{x \in \mathcal{X}_i} \|x - c_i\|^2,$$

i.e., the COST function is the total ℓ_2^2-distance between the points in $\mathcal{X}$ and their individually closest centre c_i, where $c_1, \ldots, c_k \in \mathbb{R}^d$ are chosen to minimise this distance. We further define the optimal clustering cost as

$$\Delta_k^2(\mathcal{X}) \triangleq \min_{\text{partition } \mathcal{X}_1,\ldots,\mathcal{X}_k} \mathsf{COST}(\mathcal{X}_1, \ldots, \mathcal{X}_k), \tag{4.2}$$

that is, the minimum COST of a k-way partition of $\mathcal{X}$. A k-means algorithm seeks to minimise such COST function.

As its output, Spectral Clustering returns the partition of the vertex set V corresponding to the k-means clustering computed on $\{F(u)\}_{u \in V}$: see Figure 4.1 for an illustration of the three steps of Spectral Clustering. For more background about Spectral Clustering, we refer the reader to the survey [74] and references therein.

Let's first discuss the specific choice of the spectral embedding (4.1). Observe that we normalise the eigenvectors of the Laplacian with respect to the square root of the degree of each vertex. This is equivalent to computing the eigenvectors of the *random walk matrix* $D_G^{-1} A_G$. To understand why this normalisation is needed, consider Theorem 3.1: $f_1, \ldots, f_k$ are approximated by linear combinations of the vectors $\{\overline{\chi}_{S_i}\}_{i=1}^k$, which are defined as

$$\overline{\chi}_{S_i} = \frac{D_G^{1/2} \chi_{S_i}}{\left\| D_G^{1/2} \chi_{S_i} \right\|}.$$

Therefore, considering $\{D_G^{-1/2} \overline{\chi}_{S_i}\}_{i=1}^k$ ensures that vertices from the same cluster are assigned the same value for each $D_G^{-1/2} \overline{\chi}_{S_i}$. Indeed, thanks to this normalisation and Theorem 3.1, in Lemma 4.1 we will show that vertices from the same cluster are mapped by (4.1) to points that are close to each other.

The second step of Spectral Clustering consists in clustering points in the spectral embedding with a k-means algorithm. We remark that the problem of finding an optimal k-means solution is NP-hard even when the points belong to $\mathbb{R}^2$ [43], and there is a hefty literature devoted to designing polynomial-time algorithms with provable approximation guarantees (see, e.g., [20, 33, 37, 46]). For example, [33] propose a polynomial-time algorithm for k-means that, for any constant $\varepsilon > 0$, achieves a $(9 + \varepsilon)$-approximation ratio, where we say that an algorithm for k-means achieves an APT-approximation ratio if, for any set of points $\mathcal{X} \subseteq \mathbb{R}^d$, it outputs a partition $\{A_1, \ldots, A_k\}$ of $\mathcal{X}$ such that $\mathsf{COST}(A_1, \ldots, A_k) \leq \mathsf{APT} \cdot \Delta_k^2(\mathcal{X})$. Moreover, there are

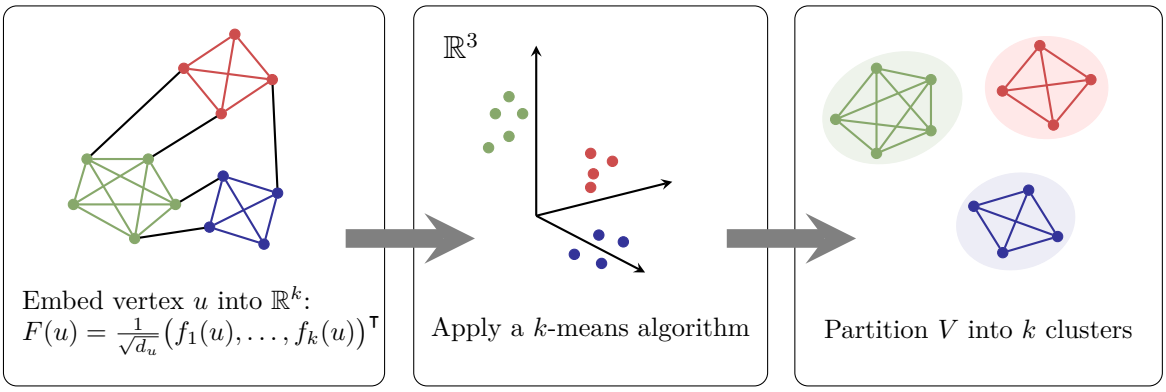


Figure 4.1: The three steps of a Spectral Clustering algorithm: (1) embed each vertex u to a point in $\mathbb{R}^k$ based on the top or bottom k eigenvectors of a matrix representing the graph; (2) apply a k-means algorithm to group the embedded points into k clusters; (3) return the corresponding partition of the vertex set as output.

simple heuristics, e.g., [41], that, although they do not provide rigorous approximation guarantees, work usually well in practice.

The widespread popularity of Spectral Clustering is due not only to its simplicity and fast runtime, but also to its surprising effectiveness in clustering real-world data (see, e.g., [73]). On the other hand, even though there are many intuitive explanations about why it works so well [74], Spectral Clustering still lacked a strong theoretical analysis prior to our work. In this chapter we study the performance of Spectral Clustering on the class of well-clustered graphs defined in Chapter 3. To the best of our knowledge, our work is the first to provide guarantees for Spectral Clustering for a large class of graphs outside stochastic models and toy examples.

The main results in this chapter are summarised in Theorem 4.1. Recall that the symmetric difference between two sets A, B is defined as $A \triangle B = (A \cup B) \setminus (A \cap B)$.

Theorem 4.1. *Let $G = (V, E, w)$ be a graph satisfying $\Upsilon_G(k) > C \cdot k^2$ for a large enough constant $C > 0$, and let $\{S_i\}_{i=1}^k$ be an optimal k-way partition of G. Let $\{A_i\}_{i=1}^k$ be a (suitably ordered) k-way partition computed by Spectral Clustering with any k-means algorithm that achieves an approximation ratio APT. Then, for any $i = 1, \ldots, k$, the following statements hold:*

1. $\mathrm{vol}(A_i \triangle S_i) = O\left(\mathsf{APT} \cdot k^2/\Upsilon_G(k)\right) \mathrm{vol}(S_i)$,
2. $\phi_G(A_i) \leq 1.1 \cdot \phi_G(S_i) + O\left(\mathsf{APT} \cdot k^2/\Upsilon_G(k)\right)$.

Notice that the quality of the approximation of the k-way expansion of the partition computed by Spectral Clustering depends on λ_{k+1}: if λ_{k+1} is a constant, i.e., the clusters are expanders, Theorem 4.1 implies Spectral Clustering achieves a constant approximation ratio of $\rho_G(k)$. This is consistent with an improved analysis of Cheeger inequality [38]. When λ_{k+1} is a function of n, however, the approximation guarantees with respect to the k-way expansion constant given by Theorem 4.1 are strictly weaker than the approximation guarantees provided by the improved analysis of Cheeger inequality. This is due to the fact that, in contrast with general k-means algorithms, the algorithm behind the higher order Cheeger inequality explicitly aims at minimising the conductance of the clusters. In particular, it might be possible that a k-means algorithm misclassifies a small fraction of the vertices breaking the conductance of a cluster, but without incurring a significant k-means cost. If this can happen, there is no hope in providing stronger guarantees for the conductance of a partition computed by Spectral Clustering.

The rest of the chapter is organised as follows. In Section 4.2 we show some interesting properties of the spectral embedding for well-clustered graphs. In Section 4.3 we exploit these properties to show that the Spectral Clustering algorithm described above can approximately recover the optimal clusters of a well-clustered graph, and the approximation guarantees of Spectral Clustering can be rigorously bounded. In Section 4.4 we show that the spectral embedding of well-clustered graphs satisfies a *separability* condition, which makes k-means clustering easier.

For the rest of the chapter, we assume there exists a large enough constant $C > 0$ such that

$$\Upsilon_G(k) > C \cdot k^2.$$

4.2 Analysis of the spectral embedding

In this section we prove some notable properties of the spectral embedding of well-clustered graphs. These properties are crucial to both the analysis of the performance of Spectral Clustering and the design of a nearly-linear time algorithm for partitioning well-clustered graphs (see Chapter 5).

The starting point of our analysis is to show there are k points $p^{(1)}, \ldots, p^{(k)} \in \mathbb{R}^k$, written as

$$p^{(i)} = \frac{1}{\sqrt{\mathrm{vol}(S_i)}} \left(\beta_i^{(1)}, \ldots, \beta_i^{(k)}\right)^\intercal \qquad i = 1, \ldots, k, \tag{4.3}$$

where the coefficients $\{\beta_i^{(j)}\}_{j=1}^k$ are defined in Theorem 3.1, such that (1) points in the spectral embedding corresponding to the vertices in a cluster S_i are concentrated around $p^{(i)}$; (2) these k points $p^{(i)}$'s are far apart from each other. These properties imply that any k-means algorithm with reasonable approximation guarantees is able to retrieve a very good approximation of the optimal clusters.

To give some intuition behind the definition of these points, consider again the example where G consists of k connected components. In this case, the bottom eigenvectors $f_1, \ldots, f_k$ of $\mathcal{L}_G$ can be written as linear combinations of the indicator vectors of the k connected components, i.e., $f_i = \beta_1^{(i)}\chi_{S_1} + \cdots + \beta_k^{(i)}\chi_{S_k}$. It follows from the definition of the χ_{S_i}'s that each vertex $u \in S_j$ is mapped to

$$F(u) = \frac{1}{\sqrt{d_u}}\left(\beta_j^{(1)}\overline{\chi}_{S_j}(u), \ldots, \beta_j^{(k)}\overline{\chi}_{S_j}(u)\right)^{\mathsf{T}} = \frac{1}{\sqrt{\mathrm{vol}(S_i)}}\left(\beta_j^{(1)}, \ldots, \beta_j^{(k)}\right)^{\mathsf{T}},$$

which is exactly the point $p^{(j)}$. By the orthogonality of the f_i's, it is also easy to show that in this example the $p^{(i)}$'s are far apart from each other. Generalising this fact we'll show that, when G is well-clustered, vertices in a cluster S_i are mapped close to $p^{(i)}$, and the $p^{(i)}$'s are far apart.

To formally prove these properties, we first show that the total ℓ_2^2-distance between points in $\mathcal{P}_i \triangleq \{F(u) : u \in S_i\}$ and $p^{(i)}$ is small. In particular, the upper bound on this distance is proportional to $1/\Upsilon_G(k)$: the more well-clustered G is, the more concentrated around $p^{(i)}$ the points in $\mathcal{P}_i$ are.

Lemma 4.1. *It holds that* $\sum_{i=1}^k \sum_{u\in S_i} d_u \left\|F(u) - p^{(i)}\right\|^2 \leq 1.1k^2/\Upsilon_G(k)$.

Proof. Let $u \in S_i$. Then, for any $1 \leq j \leq k$, by the definition of $p_j^{(i)}$ and $\widehat{\chi}_j(u)$, it holds that

$$\begin{aligned}
\sum_{i=1}^k \sum_{u\in S_i} d_u \left(F(u)_j - p_j^{(i)}\right)^2 &= \sum_{i=1}^k \sum_{u\in S_i} d_u \left(\frac{1}{\sqrt{d_u}} f_j(u) - \frac{1}{\sqrt{\mathrm{vol}(S_i)}}\beta_i^{(j)}\right)^2 \\
&= \sum_{i=1}^k \sum_{u\in S_i} \left(f_j(u) - \sqrt{\frac{d_u}{\mathrm{vol}(S_i)}}\beta_i^{(j)}\right)^2 \\
&= \sum_{i=1}^k \sum_{u\in S_i} \left(f_j(u) - \widehat{\chi}_j(u)\right)^2 \\
&= \left\|f_j - \widehat{\chi}_j\right\|^2 \\
&\leq 1.1k/\Upsilon_G(k),
\end{aligned}$$

where the last inequality follows from Theorem 3.1. Summing over all j for $1 \leq j \leq k$ implies that

$$\sum_{i=1}^{k} \sum_{u \in S_i} d_u \left\| F(u) - p^{(i)} \right\|^2 = \sum_{i=1}^{k} \sum_{j=1}^{k} \sum_{u \in S_i} d_u \left(F(u)_j - p_j^{(i)} \right)^2 \leq 1.1k^2/\Upsilon_G(k).$$

□

Observe that in Lemma 4.1 the ℓ_2^2-distances between points in $\mathcal{P}_i$ and $p^{(i)}$ are weighted by the degree of the corresponding vertices, which implies high degree vertices are more unlikely to be far from $p^{(i)}$.

To continue studying properties of the points $p^{(1)}, \ldots, p^{(k)}$ we need to introduce some notation and prove a technical lemma. Let $\beta^{(j)} \triangleq \left(\beta_1^{(j)}, \ldots, \beta_k^{(j)}\right)^\intercal$ for $1 \leq j \leq k$. Let B the $k \times k$ matrix such that $B_{i,j} = \beta_i^{(j)}$, i.e., the jth column of B is the vector $\beta^{(j)}$. The following lemma characterises the singular values of B: it essentially shows that B is almost orthogonal and the vectors $\beta^{(1)}, \ldots, \beta^{(k)}$ are almost orthonormal.

Lemma 4.2. *Let $B \in \mathbb{R}^{k \times k}$ be defined as above. All the eigenvalues of $B^\intercal B$ and $BB^\intercal$ are in the interval*

$$\left[1 - 7\sqrt{k^2/\Upsilon_G(k)}, 1 + 7\sqrt{k^2/\Upsilon_G(k)}\right].$$

Proof. First notice that the diagonal entries of $B^\intercal B$ are just the squared norm of the vectors $\beta^{(j)}$'s. By (3.9) we have that, for any $1 \leq j \leq k$

$$\left(B^\intercal B\right)_{j,j} = \left\|\beta^{(j)}\right\|^2 \leq 1 + 4k/\Upsilon_G(k). \tag{4.4}$$

Analogously, we can derive a lower bound on $\left(B^\intercal B\right)_{j,j}$. Recall that, by (3.7), for any $1 \leq i \leq k$,

$$f_i = \beta_1^{(i)} \widehat{f}_{S_1} + \cdots + \beta_k^{(i)} \widehat{f}_{S_k}.$$

Then, we have that

$$\begin{aligned}
1 = \|f_i\|^2 &= \sum_{\ell=1}^{k} \left(\beta_\ell^{(i)}\right)^2 \left\|\widehat{f}_{S_\ell}\right\|^2 + \sum_{\ell \neq \ell'} \beta_\ell^{(i)} \beta_{\ell'}^{(i)} \left\langle \widehat{f}_{S_\ell}, \widehat{f}_{S_{\ell'}} \right\rangle \\
&\leq \|\beta^{(i)}\|^2 (1 + 1/\Upsilon_G(k)) + \sum_{\ell} \left|\beta_\ell^{(i)}\right| \sum_{\ell' \neq \ell} \left|\beta_{\ell'}^{(i)}\right| \left\langle \widehat{f}_{S_\ell}, \widehat{f}_{S_{\ell'}} \right\rangle \\
&\leq \|\beta^{(i)}\|^2 (1 + 1/\Upsilon_G(k)) + \left(\sqrt{k} \cdot \|\beta^{(i)}\|\right) \cdot \left(\sqrt{k} \cdot \|\beta^{(i)}\|\right) \cdot 3/\Upsilon_G(k) \\
&\leq \left(1 + 1/\Upsilon_G(k) + 3k/\Upsilon_G(k)\right) \|\beta^{(i)}\|^2,
\end{aligned}$$

where the second equality follows from (3.7); the first inequality holds by Theorem 3.1 and since, by orthogonality, $\left\|\overline{\chi}_{S_\ell}\right\|^2 = \left\|\widehat{f}_{S_\ell}\right\|^2 + \left\|\widehat{f}_{S_\ell} - \overline{\chi}_{S_\ell}\right\|^2$; the second inequality holds by the Cauchy-Schwarz inequality and (3.6). Therefore, we have that

$$\begin{aligned}(B^\intercal B)_{j,j} = \|\beta^{(j)}\|^2 &\geq \left(1 + \frac{1}{\Upsilon_G(k)} + \frac{3k}{\Upsilon_G(k)}\right)^{-1} \\ &\geq 1 - 4k/\Upsilon_G(k). \end{aligned} \tag{4.5}$$

We now bound the entries of $B^\intercal B$ outside the main diagonal. Let's fix a row $1 \leq j \leq k$ and sum the absolute values of all the non-diagonal entries in the same column:

$$\begin{aligned}\sum_{i\neq j}\left|(B^\intercal B)_{j,i}\right| &= \sum_{i\neq j}\left|\left\langle \beta^{(i)}, \beta^{(j)}\right\rangle\right| = \sum_{i\neq j}|\langle \widehat{\chi}_i, \widehat{\chi}_j\rangle| = \sum_{i\neq j}|\langle f_i - (f_i - \widehat{\chi}_i), f_j - (f_j - \widehat{\chi}_j)\rangle| \\ &= \sum_{i\neq j}|\langle f_i, f_j\rangle - \langle f_i - \widehat{\chi}_i, f_j\rangle - \langle f_j - \widehat{\chi}_j, f_i\rangle + \langle f_i - \widehat{\chi}_i, f_j - \widehat{\chi}_j\rangle| \\ &\leq \sum_{i\neq j}\left(|\langle f_i - \widehat{\chi}_i, f_j\rangle| + |\langle f_j - \widehat{\chi}_j, f_i\rangle| + |\langle f_i - \widehat{\chi}_i, f_j - \widehat{\chi}_j\rangle|\right) \\ &\leq 3\sqrt{k^2/\Upsilon_G(k)},\end{aligned}$$

where the last line follows by Theorem 3.1 Part 2 and the Cauchy-Schwarz inequality. As done in the proof of Theorem 3.1 Part 2 we can bound the eigenvalues of $B^\intercal B$ and $BB^\intercal$ using Theorem 3.2. Let λ be an eigenvalue of $B^\intercal B$. Then, there exists $1 \leq j \leq k$ such that

$$\left|\lambda - (B^\intercal B)_{j,j}\right| \leq 3\sqrt{k^2/\Upsilon_G(k)},$$

which by (4.4) and (4.5) implies that

$$\lambda \in \left[1 - 7\sqrt{k^2/\Upsilon_G(k)}, 1 + 7\sqrt{k^2/\Upsilon_G(k)}\right].$$

Since $B^\intercal B$ and $BB^\intercal$ have the same eigenvalues, the claim holds for any eigenvalue of $BB^\intercal$ as well. □

We now show that $\left\|p^{(i)}\right\|^2$ is inversely proportional to the volume of S_i. Therefore, points from a larger cluster are mapped closer to the origin than points from smaller ones. Thanks to this property we are able to give approximation guarantees for each cluster returned by Spectral Clustering with respect to its individual volume.

Lemma 4.3. *For any $1 \leq i \leq k$, it holds that*

$$\left(1 - \frac{7k}{\sqrt{\Upsilon_G(k)}}\right) \frac{1}{\mathrm{vol}(S_i)} \leq \left\|p^{(i)}\right\|^2 \leq \left(1 + \frac{7k}{\sqrt{\Upsilon_G(k)}}\right) \frac{1}{\mathrm{vol}(S_i)}.$$

Proof. Let $\beta_i \triangleq \left(\beta_i^{(1)}, \ldots, \beta_i^{(k)}\right)^\intercal$ for any $1 \leq i \leq k$. Then, by the definition of $p^{(i)}$ we have that

$$\left\|p^{(i)}\right\|^2 = \frac{1}{\mathrm{vol}(S_i)} \left\|\beta_i\right\|^2. \tag{4.6}$$

To bound $\|\beta_i\|^2$, notice that $\|\beta_i\|^2 = (BB^\intercal)_{i,i}$. Lemma 4.2 states that all the eigenvalues of $BB^\intercal$ are in the interval $\left[1 - 7\sqrt{k^2/\Upsilon_G(k)}, 1 + 7\sqrt{k^2/\Upsilon_G(k)}\right]$. By the spectral theorem, we write $BB^\intercal$ as $BB^\intercal = \sum_{j=1}^k \lambda_j v^{(j)} v^{(j)\intercal}$ where the λ_j's are the eigenvalues of $BB^\intercal$ and the $v^{(j)}$'s the corresponding orthonormal eigenvectors. This implies that

$$\|\beta_i\|^2 = (BB^\intercal)_{i,i} = \sum_{j=1}^k \lambda_j \left(v_i^{(j)}\right)^2 \in \left[1 - \frac{7k}{\sqrt{\Upsilon_G(k)}}, 1 + \frac{7k}{\sqrt{\Upsilon_G(k)}}\right] \sum_{j=1}^k \left(v_i^{(j)}\right)^2.$$

But since the $v^{(j)}$'s are orthonormal and span the whole $\mathbb{R}^k$, we have that $\sum_{j=1}^k v^{(j)} v^{(j)\intercal} = \mathbb{I}$ and

$$\|\beta_i\|^2 = (BB^\intercal)_{i,i} \in \left[1 - \frac{7k}{\sqrt{\Upsilon_G(k)}}, 1 + \frac{7k}{\sqrt{\Upsilon_G(k)}}\right],$$

Combining this with (4.6) proves the lemma. □

We now show that the $p^{(i)}$'s are far apart from each other. More precisely, we show that the distance between $p^{(i)}$ and $p^{(j)}$ is inversely proportional to the minimum of the volume of S_i and S_j.

Lemma 4.4. *Let $i \neq j$. Then, it holds that*

$$\begin{aligned}\left\|p^{(i)} - p^{(j)}\right\|^2 &\geq \left(1 - \frac{21k}{\sqrt{\Upsilon_G(k)}}\right) \frac{1}{\min\{\mathrm{vol}(S_i), \mathrm{vol}(S_j)\}} \\ &\geq \frac{1}{2\min\{\mathrm{vol}(S_i), \mathrm{vol}(S_j)\}}.\end{aligned}$$

Proof. We first bound the inner product between $p^{(i)}$ and $p^{(j)}$. By (4.3), we have that

$$\left|\left\langle p^{(i)}, p^{(j)}\right\rangle\right| = \frac{1}{\sqrt{\mathrm{vol}(S_i) \cdot \mathrm{vol}(S_j)}} \left|\left(\beta_i^{(1)}, \ldots, \beta_i^{(k)}\right)\left(\beta_j^{(1)}, \ldots, \beta_j^{(k)}\right)^\intercal\right|. \tag{4.7}$$

Notice that $\left(\beta_i^{(1)}, \ldots, \beta_i^{(k)}\right)^\intercal \left(\beta_j^{(1)}, \ldots, \beta_j^{(k)}\right)$ is just the entry in the ith row and jth column of the matrix $BB^\intercal$. Similar to the proof of Lemma 4.3, we consider the spectral decomposition $BB^\intercal = \sum_{j=1}^k \mu_j v^{(j)} v^{(j)\intercal}$. By Lemma 4.2 we have that

$$\begin{aligned}(BB^\intercal)_{i,j} = \sum_{\ell=1}^k \mu_\ell \cdot v_i^{(\ell)} v_j^{(\ell)} \leq \left(1 + \frac{7k}{\sqrt{\Upsilon_G(k)}}\right) \sum_{\ell=1}^k v_i^{(\ell)} v_j^{(\ell)} \mathbb{1}\left\{v_i^{(\ell)} v_j^{(\ell)} > 0\right\} \\ + \left(1 - \frac{7k}{\sqrt{\Upsilon_G(k)}}\right) \sum_{\ell=1}^k v_i^{(\ell)} v_j^{(\ell)} \mathbb{1}\left\{v_i^{(\ell)} v_j^{(\ell)} < 0\right\}.\end{aligned}$$

By the orthonormality of the $v^{(j)}$'s and the fact that they span the whole $\mathbb{R}^k$, we have that $\sum_{\ell=1}^k v^{(\ell)} v^{(\ell)\intercal} = \mathbb{I}$ and $\sum_{\ell=1}^k v_i^{(\ell)} v_j^{(\ell)} = 0$, which implies

$$\sum_{\ell=1}^k v_i^{(\ell)} v_j^{(\ell)} \mathbb{1}\left\{v_i^{(\ell)} v_j^{(\ell)} > 0\right\} = \sum_{\ell=1}^k v_i^{(\ell)} v_j^{(\ell)} \mathbb{1}\left\{v_i^{(\ell)} v_j^{(\ell)} < 0\right\}.$$

Hence,

$$(BB^\intercal)_{i,j} \leq \frac{14k}{\sqrt{\Upsilon_G(k)}} \sum_{\ell=1}^k \left|v_i^{(\ell)} v_j^{(\ell)}\right| \leq \frac{14k}{\sqrt{\Upsilon_G(k)}},$$

where the last inequality follows from the fact that $\sum_{\ell=1}^k \left|v_i^{(\ell)}\right|^2 = \sum_{\ell=1}^k \left|v_j^{(\ell)}\right|^2 = 1$.

Analogously, we can also prove that $(BB^\intercal)_{i,j} \geq -14k/\sqrt{\Upsilon_G(k)}$. By (4.7) we have that

$$\left|\langle p^{(i)}, p^{(j)} \rangle\right| \leq \frac{14k}{\sqrt{\Upsilon_G(k)}} \cdot \frac{1}{\sqrt{\mathrm{vol}(S_i) \cdot \mathrm{vol}(S_j)}}.$$

Therefore, by Lemma 4.3, it holds that

$$\begin{aligned}\left\|p^{(i)} - p^{(j)}\right\|^2 &= \left\|p^{(i)}\right\|^2 + \left\|p^{(j)}\right\|^2 - 2\left\langle p^{(i)}, p^{(j)} \right\rangle \\ &\geq \left(1 - \frac{7k}{\sqrt{\Upsilon_G(k)}}\right) \frac{1}{\mathrm{vol}(S_i)} + \left(1 - \frac{7k}{\sqrt{\Upsilon_G(k)}}\right) \frac{1}{\mathrm{vol}(S_j)} \\ &\quad - \frac{14k}{\sqrt{\Upsilon_G(k)}} \cdot \frac{1}{\sqrt{\mathrm{vol}(S_i) \cdot \mathrm{vol}(S_j)}} \\ &\geq \left(1 - \frac{28k}{\sqrt{\Upsilon_G(k)}}\right) \frac{1}{\min\{\mathrm{vol}(S_i), \mathrm{vol}(S_j)\}},\end{aligned}$$

which proves the first inequality in the statement of the lemma. The second inequality follows by the assumption $\Upsilon_G(k) = \Omega(k^2)$. □

Before we apply these properties to obtain approximation guarantees for Spectral Clustering algorithms, it is worth summarising these properties. We have shown there exist k points $p^{(1)}, \ldots, p^{(k)}$ such that each cluster S_i is mapped by the spectral embedding to points that are concentrated around the corresponding $p^{(i)}$. In Lemma 4.1 we have proved, as we would expect, that a higher value of $\Upsilon_G(k)$ corresponds to a higher degree of concentration. Moreover, these points $p^{(i)}$'s lie in almost orthogonal directions (Lemma 4.4) and their squared norms are inversely proportional to the volumes of their corresponding cluster (Lemma 4.3). This means embedded points from a larger cluster enjoy a higher degree of concentration around their corresponding point $p^{(i)}$ than points from smaller clusters. Indeed, suppose for the sake of the argument that this is not the case and the distance between points in the embedding and their respective $p^{(i)}$ is uniform. Then, when an algorithm tries to minimise the squared distances between points in the same cluster, as k-means does, obtaining a bad approximation of small clusters wouldn't significantly affect the total k-means cost. On the contrary, thanks to this property, Spectral Clustering is able to recover a good approximation of all the clusters, no matter how small the size of a cluster is. This fact is illustrated in Figure 4.2 and will be formally proved in the next section.

4.3 Approximation guarantees

In this section we provide approximation guarantees for Spectral Clustering. We assume that $A_1, \ldots, A_k$ is a k-way partition computed by a k-means algorithm on the point-set $\{F(u)\}_{u \in V}$ and define the cost of $A_1, \ldots, A_k$ as

$$\mathsf{COST}(A_1, \ldots, A_k) \triangleq \min_{c_1, \ldots, c_k \in \mathbb{R}^k} \sum_{i=1}^{k} \sum_{u \in A_i} d_u \|F(u) - c_i\|^2.$$

We further define the optimal clustering cost as

$$\Delta_k^2 \triangleq \min_{\text{partition } A_1, \ldots, A_k} \mathsf{COST}(A_1, \ldots, A_k).$$

i.e., we define the optimal clustering cost in the same way as in (4.2), except that we look at the embedded points from vertices of G in the definition, and we count

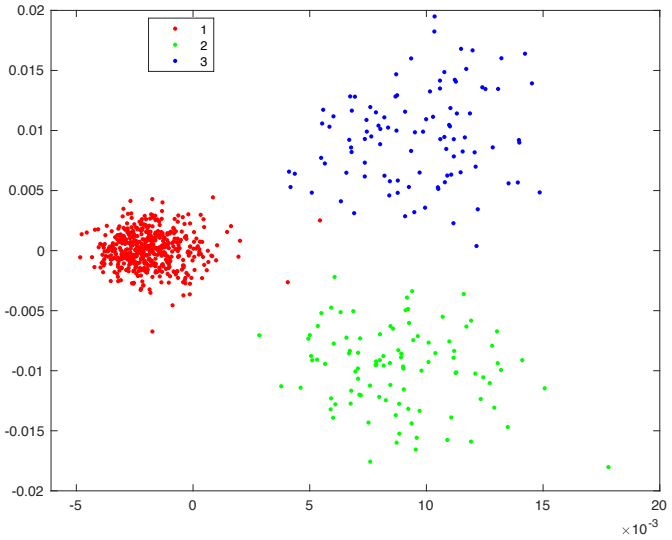


Figure 4.2: Plot of the spectral embedding of a random graph with 3 clusters consisting of 500, 100, and 100 vertices respectively. Vertices within the same cluster are connected by an edge with probability 0.3, and vertices belonging to different clusters are connected with probability 0.1. The two dimensions correspond to the values of the second and third eigenvectors of the Laplacian of the graph, normalised by the degrees of the vertices, as in (4.1). We omit the first coordinate of the spectral embedding since it is always constant. Notice how points in the largest (red) cluster have a higher degree of concentration and are closer to the origin.

the distance between a point $F(u)$ and its closest centre d_u times. Notice that we can apply any approximation algorithm for k-means to minimise this cost function simply mapping each vertex u to d_u identical points $F(u) \in \mathbb{R}^k$. Giving more weight to high degree vertices, we will be able to bound the volume of the overlap between the clusters retrieved by a k-means algorithm and the optimal ones. From now on, we always refer to COST and Δ_k^2 as the COST and optimal COST values of the points $\{F(u)\}_{u \in V}$, where every point $F(u)$ is counted d_u times. We further assume that the k-means algorithm employed achieves an APT-approximation ratio, that is, for any set of points $\mathcal{X} \subseteq \mathbb{R}^d$, it outputs a partition $\{A_1, \ldots, A_k\}$ of $\mathcal{X}$ such that

$$\mathsf{COST}(A_1, \ldots, A_k) \leq \mathsf{APT} \cdot \Delta_k^2(\mathcal{X}).$$

Lemma 4.5 below bounds the cost of the optimal k-means clustering with respect to the value of $\Upsilon_G(k)$.

Lemma 4.5. *It holds that* $\Delta_k^2 \leq 1.1k^2/\Upsilon_G(k)$.

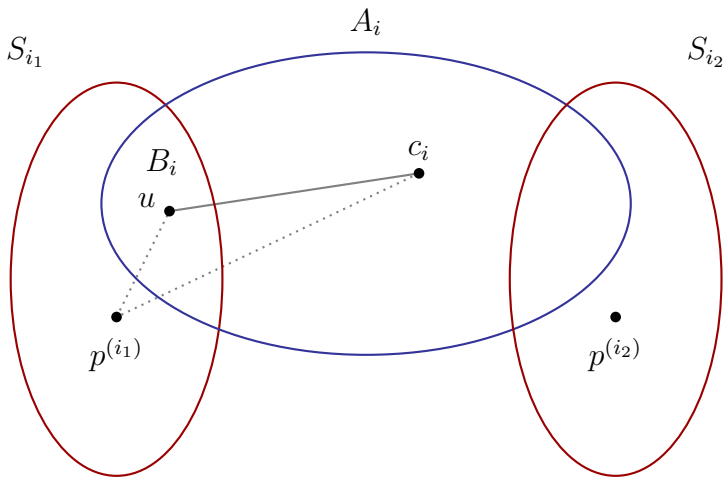


Figure 4.3: We use the fact that $\left\|p^{(i_1)} - c_i\right\| \geq \left\|p^{(i_2)} - c_i\right\|$ to lower bound the value of COST function by only looking at the contribution of points $u \in B_i$ for all $1 \leq i \leq k$.

Proof. Since Δ_k^2 is obtained by minimising over all partitions $A_1, \ldots, A_k$ and $c_1, \ldots, c_k$, we have that

$$\Delta_k^2 \leq \sum_{i=1}^{k} \sum_{u \in S_i} d_u \left\|F(u) - p^{(i)}\right\|^2. \tag{4.8}$$

Hence the statement follows by applying Lemma 4.1. □

Since $A_1, \cdots, A_k$ is the output of a k-means algorithm with approximation ratio APT, by Lemma 4.5 we have that $\mathsf{COST}(A_1, \ldots, A_k) \leq \mathsf{APT} \cdot 1.1k^2/\Upsilon_G(k)$. We will show that this upper bound implies that $A_1, \ldots, A_k$ is close to the optimal clustering $S_1, \ldots, S_k$, in the sense that, for some permutation $\sigma : \{1, \ldots, k\} \to \{1, \ldots, k\}$, the symmetric difference between A_i and $S_{\sigma(i)}$ is small. In particular Lemma 4.6 shows that if the symmetric difference between some A_i and its correspondence $S_{\sigma(i)}$ is large, then there exists a cluster A_j that contains a large fraction of the volume of two different clusters S_{j_1} and S_{j_2}. But since S_{j_1} and S_{j_2} are far apart in the spectral embedding, $\mathsf{COST}(A_1, \ldots, A_k)$ would be large, contradicting Lemma 4.5.

Lemma 4.6. *Let $A_1, \ldots, A_k$ be a partition of V. Suppose that, for every permutation of the indices $\sigma : \{1, \ldots, k\} \to \{1, \ldots, k\}$, there exists $i^\star$ such that*

$$\operatorname{vol}\left(A_{i^\star} \triangle S_{\sigma(i^\star)}\right) \geq 2\varepsilon \operatorname{vol}\left(S_{\sigma(i^\star)}\right)$$

for $1/2 \geq \varepsilon \geq 50(1+\mathsf{APT}) \cdot k^2/\Upsilon_G(k)$, then $\mathsf{COST}(A_1, \ldots, A_k) \geq 1.2k^2(1+\mathsf{APT})/\Upsilon_G(k)$.

Proof. We first consider the case where there exists a permutation $\sigma : \{1, \dots, k\} \to \{1, \dots, k\}$ such that, for any $1 \leq j \leq k$,

$$\operatorname{vol}\left(A_j \cap S_{\sigma(j)}\right) > \frac{1}{2} \operatorname{vol}\left(S_{\sigma(j)}\right). \tag{4.9}$$

This assumption essentially says that $A_1, \dots, A_k$ is a non-trivial approximation of the optimal clustering $S_1, \dots, S_k$ according to some permutation σ. Later we will show that if (4.9) is not satisfied, the statement of the Lemma trivially holds.

From the hypothesis of the Lemma, there is $1 \leq i^\star \leq k$ such that $\operatorname{vol}\left(A_{i^\star} \triangle S_{\sigma(i^\star)}\right) \geq 2\varepsilon \operatorname{vol}\left(S_{\sigma(i^\star)}\right)$ for some $1/2 \geq \varepsilon \geq 50(1 + \mathsf{APT}) \cdot k^2/\Upsilon_G(k)$. By the definition of symmetric difference of two sets, one of the following two cases must hold:

1. A large portion of $A_{i^\star}$ belongs to clusters different from $S_{\sigma(i^\star)}$, i.e., there exist $\varepsilon_1, \dots, \varepsilon_k \geq 0$ such that $\varepsilon_{i^\star} = 0$, $\sum_{j=1}^k \varepsilon_j \geq \varepsilon$, and $\operatorname{vol}\left(A_{i^\star} \cap S_{\sigma(j)}\right) \geq \varepsilon_j \operatorname{vol}\left(S_{\sigma(i^\star)}\right)$ for any $1 \leq j \leq k$.

2. $A_{i^\star}$ is missing a large portion of $S_{\sigma(i^\star)}$, which must have been assigned to other clusters. Therefore, we can define $\varepsilon_1, \dots, \varepsilon_k \geq 0$ such that $\varepsilon_{i^\star} = 0$, $\sum_{j=1}^k \varepsilon_j \geq \varepsilon$, and $\operatorname{vol}\left(A_j \cap S_{\sigma(i^\star)}\right) \geq \varepsilon_j \operatorname{vol}\left(S_{\sigma(i^\star)}\right)$ for any $1 \leq j \leq k$.

In both cases, we can define sets $B_1, \dots, B_k$ and $D_1, \dots, D_k$ such that B_j and D_j belong to the same cluster $A_{j'}$ but to two different optimal clusters S_{j_1} and S_{j_2}. Hence, points in B_j and D_j must be mapped to regions of the spectral embedding far apart from each other, implying a high k-means cost for $A_1, \dots, A_k$. More precisely, in the first case, for any $1 \leq j \leq k$, we define $B_j = A_{i^\star} \cap S_{\sigma(j)}$. We define $D_1, \dots, D_j$ as an arbitrarily partition of $A_{i^\star} \cap S_{\sigma(i^\star)}$ with the constraint that $\operatorname{vol}(D_j) \geq \varepsilon_j \operatorname{vol}(S_{\sigma(i^\star)})$. This is possible since by (4.9) $\operatorname{vol}(A_{i^\star} \cap S_{\sigma(i^\star)}) \geq \frac{1}{2} \operatorname{vol}(S_{\sigma(i^\star)}) \geq \varepsilon \operatorname{vol}(S_{\sigma(i^\star)})$. In the second case, instead, for any $1 \leq j \leq k$, we define $B_j = A_j \cap S_{\sigma(i^\star)}$ and $D_j = A_j \cap S_{\sigma(j)}$. Note that it also holds by (4.9) that $\operatorname{vol}(D_j) \geq \varepsilon_j \operatorname{vol}(S_{\sigma(j)})$. We can then combine the two cases together (albeit using different definitions for the sets) and assume that there exist $\varepsilon_1, \dots, \varepsilon_k \geq 0$ such that $\varepsilon_{i^\star} = 0$, $\sum_{j=1}^k \varepsilon_j \geq \varepsilon \geq 50(1 + \mathsf{APT}) \cdot k^2/\Upsilon_G(k)$, and such that we can find collections of pairwise disjoint sets $\{B_1, \dots, B_k\}$ and $\{D_1, \dots, D_k\}$ with the following properties: for any $j \neq i$ there exist indices j' and $j_1 \neq j_2$ such that

1. $B_j, D_j \subseteq A_{j'}$
2. $D_j \subseteq S_{j_1}, B_j \subseteq S_{j_2}$
3. $\operatorname{vol}(B_j) \geq \varepsilon_j \min\{\operatorname{vol}(S_{j_i}), \operatorname{vol}(S_{j_2})\}$

4. $\mathrm{vol}(D_j) \geq \varepsilon_j \min\{\mathrm{vol}\left(S_{j_i}\right), \mathrm{vol}\left(S_{j_2}\right)\}$

For any j, we define c_j as the centre of the corresponding cluster $A_{j'}$ to which both B_j and D_j are subset of. We can also assume without loss of generality that $\|c_j - p^{(j_1)}\| \geq \|c_j - p^{(j_2)}\|$ which implies $\|p^{(j_1)} - c_j\| \geq \|p^{(j_1)} - p^{(j_2)}\|/2$. As a consequence, points in B_j are far away from c_j (see Figure 4.3 for an illustration). Notice that if instead $\|c_j - p^{(j_1)}\| < \|c_j - p^{(j_2)}\|$, we would just need to reverse the role of B_j and D_j without changing the proof. We now bound $\mathsf{COST}(A_1, \ldots, A_k)$ by looking only at the contribution of the points in the B_j's. Indeed, by Lemma 4.1 the sum of the squared-distances between points in B_j and $p^{(j_1)}$ is at most $1.1k^2/\Upsilon_G(k)$, while the distance between $p^{(j_1)}$ and $p^{(j_2)}$ is large (Lemma 4.4). Therefore, we have that

$$\mathsf{COST}(A_1, \ldots, A_k) = \sum_{j=1}^{k} \sum_{u \in A_j} d_u \|F(u) - c_j\|^2 \geq \sum_{j=1}^{k} \sum_{u \in B_j} d_u \|F(u) - c_j\|^2.$$

By applying the inequality $a^2 + b^2 \geq (a-b)^2/2$, we have that

$$\begin{aligned}
\mathsf{COST}(A_1, \ldots, A_k) &\geq \sum_{j=1}^{k} \sum_{u \in B_j} d_u \left(\frac{\left\|p^{(j_1)} - c_j\right\|^2}{2} - \left\|F(u) - p^{(j_1)}\right\|^2 \right) \\
&\geq \sum_{j=1}^{k} \sum_{u \in B_j} d_u \frac{\left\|p^{(j_1)} - c_j\right\|^2}{2} - \sum_{j=1}^{k} \sum_{u \in B_j} d_u \left\|F(u) - p^{(j_1)}\right\|^2 \\
&\geq \sum_{j=1}^{k} \sum_{u \in B_j} d_u \frac{\left\|p^{(j_1)} - c_j\right\|^2}{2} - \frac{1.1k^2}{\Upsilon_G(k)} \qquad (4.10) \\
&\geq \sum_{j=1}^{k} \sum_{u \in B_j} d_u \frac{\left\|p^{(j_1)} - p^{(j_2)}\right\|^2}{8} - \frac{1.1k^2}{\Upsilon_G(k)} \\
&\geq \sum_{j=1}^{k} \frac{\mathrm{vol}(B_j)}{16 \min\{\mathrm{vol}(S_{j_1}), \mathrm{vol}(S_{j_2})\}} - \frac{1.1k^2}{\Upsilon_G(k)} \qquad (4.11) \\
&\geq \sum_{j=1}^{k} \frac{\varepsilon_i \min\{\mathrm{vol}(S_{j_1}), \mathrm{vol}(S_{j_2})\}}{16 \min\{\mathrm{vol}(S_{j_1}), \mathrm{vol}(S_{j_2})\}} - \frac{1.1k^2}{\Upsilon_G(k)} \\
&\geq \sum_{j=1}^{k} \frac{\varepsilon_i}{16} - \frac{1.1k^2}{\Upsilon_G(k)} \\
&\geq \frac{\varepsilon}{16} - \frac{1.1k^2}{\Upsilon_G(k)} \qquad (4.12) \\
&\geq \frac{1.2(1 + \mathsf{APT})k^2}{\Upsilon_G(k)}
\end{aligned}$$

where (4.10) follows from Lemma 4.1, (4.11) from Lemma 4.4 and the last inequality from the assumption that $\varepsilon \geq 50(1+\mathsf{APT})k^2/\Upsilon_G(k)$.

It remains to show that removing assumption (4.9) implies the Lemma as well. Notice that if (4.9) is not satisfied, for all permutations σ there exists $1 \leq \ell^\star \leq k$ such that $\operatorname{vol}\left(A_{\ell^\star} \cap S_{\sigma(\ell^\star)}\right) \leq \frac{1}{2}\operatorname{vol}\left(S_{\sigma(\ell^\star)}\right)$. We can also assume the following stronger condition:

$$\operatorname{vol}\left(A_{\ell^\star} \cap S_j\right) \leq \frac{1}{2}\operatorname{vol}\left(S_j\right) \qquad \text{for any } 1 \leq j \leq k. \tag{4.13}$$

Indeed, if there would exist a unique $j \neq \sigma(\ell^\star)$ such that $\operatorname{vol}\left(A_{\ell^\star} \cap S_j\right) > \frac{1}{2}\operatorname{vol}\left(S_j\right)$, then it would just mean that σ is the "wrong" permutation and we should consider only permutations $\sigma' \neq \sigma$ such that $\sigma'(\ell^\star) = j$. If instead there would exist $j_1 \neq j_2$ such that $\operatorname{vol}\left(A_{\ell^\star} \cap S_{j_1}\right) > \frac{1}{2}\operatorname{vol}\left(S_{j_1}\right)$ and $\operatorname{vol}\left(A_{\ell^\star} \cap S_{j_2}\right) > \frac{1}{2}\operatorname{vol}\left(S_{j_2}\right)$, then it is easy to see that the Lemma would hold, since in this case $A_{\ell^\star}$ would contain large portions of two different optimal clusters, and, as clear from the previous part of the proof, this would imply a high k-means cost.

Therefore, we just need to show that the statement of the Lemma holds when (4.13) is satisfied. For this purpose we define sets $C_1, \ldots, C_k$ which are subsets of vertices in $S_1, \ldots, S_k$ that are close in the spectral embedding to $p^{(1)}, \ldots, p^{(k)}$. Formally, for any $1 \leq j \leq k$,

$$C_j = \left\{ u \in S_j : \|F(u) - p^{(j)}\|^2 \leq \frac{1000k^2}{\Upsilon_G(k) \cdot \operatorname{vol}(S_j)} \right\}.$$

Notice that by Lemma 4.5 $\operatorname{vol}(C_j) \geq \frac{99}{100}\operatorname{vol}(S_j)$. By assumption (4.13), roughly half of the volume of all the C_j's must be contained in at most $k-1$ sets (all the A_j's different from $A_{\ell^\star}$). We prove this implies that the k-means cost is high, from which the Lemma follows.

Let $c_1, \ldots, c_k$ be the centres of $A_1, \ldots, A_k$. We are trying to assign a large portion of each of the k optimal clusters to only $k-1$ centres (namely all the centres different from $c_{\ell^\star}$). Moreover, by Lemma 4.4 any centre $c_j \neq c_{\ell^\star}$ can either be close to $p^{(\ell^\star)}$ or to another optimal centre $p^{(j')}$, but not to both. As a result, there will be at least one C_j whose points are assigned to a centre which is at least $\Omega(1/\operatorname{vol}(S_j))$ far from $p^{(j)}$ (in squared Euclidean distance). Therefore, by the definition of C_j and the fact that $\operatorname{vol}(C_j) \geq \frac{99}{100}\operatorname{vol}(S_j)$, the k-means cost is at least $\Omega\left(\frac{1}{\operatorname{vol}(S_j)} \cdot \operatorname{vol}(C_j)\right) = \Omega(1)$. This concludes the proof. □

We can now prove Theorem 4.1, which shows that Spectral Clustering computes a good approximation of the optimal partition $S_1, \ldots, S_k$. The quality of the approxima-

tion depends on the approximation guarantees achieved by the k-means algorithms used and on the value of $\Upsilon_G(k)$. The proof is based on a contradiction argument: suppose the partition computed by Spectral Clustering differs significantly from the optimal one. Then, by Lemma 4.6, the k-means cost of the partitioned computed must be high. But this contradicts the facts that we used an APT-approximation algorithm for k-means and that the k-means cost of the optimal partition must be small by Lemma 4.5.

Proof of Theorem 4.1. Let $A_1, \ldots, A_k$ be a k-way partition that achieves an approximation ratio of APT, and let $\varepsilon = 100 \cdot k^2 \cdot (1 + \mathsf{APT})/\Upsilon_G(k)$. We first show that there exists a permutation σ of the indices such that

$$\mathrm{vol}\left(A_i \triangle S_{\sigma(i)}\right) \leq \varepsilon \,\mathrm{vol}(S_{\sigma(i)}), \qquad \text{for any } 1 \leq i \leq k. \tag{4.14}$$

Assume for contradiction that for any permutation σ there is $1 \leq i \leq k$ such that $\mathrm{vol}(A_i \triangle S_{\sigma(i)}) > \varepsilon \,\mathrm{vol}\left(S_{\sigma(i)}\right)$. This implies by Lemma 4.6 that $\mathsf{COST}(A_1, \ldots, A_k) \geq 1.2 \cdot (1 + \mathsf{APT}) \cdot k^2/\Upsilon_G(k)$, which contradicts to the fact that $A_1, \ldots, A_k$ is an APT-approximation to a k-way partition, whose corresponding k-means cost is at most $1.1 \cdot \mathsf{APT} \cdot k^2/\Upsilon_G(k)$.

Now we assume that $\sigma : \{1, \cdots, k\} \to \{1, \cdots, k\}$ is the permutation satisfying (4.14), and bound the conductance of every cluster A_i. For any $1 \leq i \leq k$, the number of leaving edges of A_i is upper bounded by

$$|\partial A_i| \leq \left|\partial\left(A_i \setminus S_{\sigma(i)}\right)\right| + \left|\partial\left(A_i \cap S_{\sigma(i)}\right)\right| \leq \left|\partial\left(A_i \triangle S_{\sigma(i)}\right)\right| + \left|\partial\left(A_i \cap S_{\sigma(i)}\right)\right|.$$

Notice that $\left|\partial\left(A_i \triangle S_{\sigma(i)}\right)\right| \leq \varepsilon \,\mathrm{vol}\left(S_{\sigma(i)}\right)$ by our assumption on σ, and every node in $\left|\partial\left(A_i \cap S_{\sigma(i)}\right)\right|$ either belongs to $\left(\partial S_{\sigma(i)}\right) \setminus S_{\sigma(i)}$ or $\partial\left(A_i \triangle S_{\sigma(i)}\right)$. Therefore,

$$\begin{aligned} |\partial A_i| &\leq \varepsilon \,\mathrm{vol}\left(S_{\sigma(i)}\right) + \phi_G\left(S_{\sigma(i)}\right) \mathrm{vol}\left(S_{\sigma(i)}\right) + \varepsilon \,\mathrm{vol}\left(S_{\sigma(i)}\right) \\ &= \left(2\varepsilon + \phi_G\left(S_{\sigma(i)}\right)\right) \mathrm{vol}(S_{\sigma(i)}). \end{aligned}$$

On the other hand, we have that $\mathrm{vol}\left(A_i\right) \geq \mathrm{vol}\left(A_i \cap S_{\sigma(i)}\right) \geq (1 - 2\varepsilon)\,\mathrm{vol}(S_{\sigma(i)})$. Hence,

$$\begin{aligned} \phi_G(A_i) &\leq \frac{(2\varepsilon + \phi_G(S_{\sigma(i)}))\,\mathrm{vol}(S_{\sigma(i)})}{(1 - 2\varepsilon)\,\mathrm{vol}(S_{\sigma(i)})} = \frac{2\varepsilon + \phi_G(S_{\sigma(i)})}{1 - 2\varepsilon} \\ &\leq 1.1 \cdot \phi_G(S_{\sigma(i)}) + O(\mathsf{APT} \cdot k^2/\Upsilon_G(k)). \end{aligned}$$

□

4.4 Separability of the spectral embedding

We already mentioned that, although k-means is NP-hard in the worst case [43], there are many approximation algorithms for k-means that achieve a bounded approximation ratio (see, e.g., [33, 20, 13, 46]), and run in time polynomial in n, the number of points, k, the number of clusters, and d, the dimension of the points. Moreover, when the set of points in input satisfies some clusterability assumption, for example when the centres of the clusters are far apart from each other compared with the average distance between points in the same cluster, simple heuristics, which usually don't have rigorous theoretical guarantees in the worst case, perform extremely well and can be rigorously analysed [4]. For example, Ostrovsky et al. [52] introduce a notion of separability between the clusters and show that, when the point-set in input satisfies this condition, a simple modification of Lloyd's heuristics [41] achieves a bounded approximation ratio for k-means. More precisely, we say that a set of points $\mathcal{X}$ is separated if the k-means cost $\Delta_k^2(\mathcal{X})$ of partitioning $\mathcal{X}$ in k clusters is much smaller than the cost $\Delta_{k-1}^2(\mathcal{X})$ of partitioning $\mathcal{X}$ in $k-1$ clusters. In particular, we say that a set of points $\mathcal{X}$ is ε-separated for k-means if $\Delta_k^2(\mathcal{X}) \leq \varepsilon \cdot \Delta_{k-1}^2(\mathcal{X})$

It was shown in [52] that, if $\mathcal{X}$ is ε-separated for a small enough ε, there exists a simple algorithm that achieves a constant approximation ratio for k-means and runs in time linear in the number of points n and their dimension d, and polynomial in the number of clusters k. This result is summarised as follows.

Theorem 4.2 ([52])**.** *Suppose that $\mathcal{X}$ is ε-separated for k-means for a small enough ε. Then, there exists an algorithm that returns a solution of cost at most $(1+O(\varepsilon))\Delta_k^2(\mathcal{X})$ with probability $1-O(\varepsilon^{1/4})$ in time $O(nkd+k^3d)$.*

Now we show that we can apply Theorem 4.2 to the spectral embedding of a well-clustered graph, i.e., we prove that such embedding is ε-separated for a small ε.

Lemma 4.7. *Let $\Upsilon_G(k) > C \cdot k^2$ for some large enough constant $C > 0$, and $\mathcal{X} = \{F(u)\}_{u \in V}$ be the spectral embedding of $G = (V, E, w)$. Then, $\mathcal{X}$ is $(100/C)$-separated for k-means.*

Proof. By Lemma 4.5 we know that $\Delta_k^2(\mathcal{X}) \leq 1.1k^2/\Upsilon_G(k)$. We now study the COST value when $\mathcal{X}$ is partitioned in $k-1$ clusters. Suppose $A_1, \ldots, A_{k-1}$ is the partition of $\mathcal{X}$ in $k-1$ subsets achieving optimal cost $\Delta_{k-1}^2(\mathcal{X})$. Consider now the k-way partition $A_1, \ldots, A_k$ of $\mathcal{X}$ where $A_k = \emptyset$. By construction, $\Delta_k^2(\mathcal{X}) \leq \mathsf{COST}(A_1, \ldots, A_k) = \Delta_{k-1}^2(\mathcal{X})$. Clearly, since $A_k = \emptyset$, for any $i = 1, \ldots, k$, we have that

$\mathrm{vol}(A_k \triangle S_i) \geq \mathrm{vol}(S_i)$, where $\{S_1, \ldots, S_k\}$ is a k-way partition of V achieving $\rho_G(k)$. By the same argument of Lemma 4.6, we have that $\Delta^2_{k-1}(\mathcal{X}) = \mathsf{COST}(A_1, \ldots, A_k) \geq 1/50$. Hence,

$$\frac{\Delta^2_{k-1}(\mathcal{X})}{\Delta^2_k(\mathcal{X})} \geq \frac{\Upsilon_G(k)}{100k^2} \geq \frac{C}{100}.$$

Therefore, $\mathcal{X}$ is $(100/C)$-separated for k-means. □

Lemma 4.7 shows that, as long as $\Upsilon_G(k) \gg k^2$, we can apply Theorem 4.2 to obtain a constant-factor approximation to the k-means cost of the spectral embedding. Moreover, combining the power method with nearly-linear time Laplacian solvers [66] as explained in [71], we can compute the bottom k eigenvectors of $\mathcal{L}_G$ in $\widetilde{O}(mk)$ time, where m is the number of edges in the graph and $\widetilde{O}(\cdot)$ hides polylogarithmic factors in m. In summary, as a corollary of Theorem 4.1, we obtain the following.

Corollary 4.1. *Let $G = (V, E, w)$ be a graph of m edges satisfying the condition $\Upsilon_G(k) > C \cdot k^2$ for some large enough constant $C > 0$,, and let $\{S_i\}_{i=1}^k$ be an optimal k-way partitioning of G. There exists an algorithm that runs in time $\widetilde{O}(mk + \mathrm{poly}(k))$ and computes a k-way partition $\{A_i\}_{i=1}^k$ of V such that, for any $i = 1, \ldots, k$, the following statements hold:*

1. $\mathrm{vol}(A_i \triangle S_i) = O\left(k^2/\Upsilon_G(k)\right) \mathrm{vol}(S_i)$,
2. $\phi_G(A_i) \leq 1.1 \cdot \phi_G(S_i) + O\left(k^2/\Upsilon_G(k)\right)$.

Chapter 5

Graph clustering in nearly-linear time

In Chapter 4 we gave theoretical guarantees on the partition returned by Spectral Clustering when the input is a well-clustered graph, and show that Spectral Clustering can be implemented in time nearly linear in the number of edges in the graph m and in the number of clusters k. Since the value of k in certain applications is not always a constant, a natural question is to study if a linear time algorithm for graph clustering exists.

Before answering this question, observe that in order to design a linear time algorithm we need to overcome two barriers: (1) "writing down" the spectral embedding (4.1) requires at least $\Omega(nk)$ time; (2) $\Omega(nk)$ time is also required by most k-means algorithms. Notice that when k is large, e.g., $k = n^{0.1}$, this may not be linear in m anymore. To overcome the first obstacle, we show that the heat-kernel embedding [59] closely approximates the spectral embedding in well-clustered graphs. Combining the heat-kernel embedding with Johnson-Lindenstrauss random projections [3, 31], we are able to compute in nearly-linear time an embedding of the vertices of the graph in $O(\operatorname{poly}\log n)$ dimensions such that the pairwise distances between the points in the embedding are close to the corresponding pairwise distances in the spectral embedding. For the second obstacle, instead, we take advantage of the specific properties of the spectral embedding of well-clustered graphs to obtain an ad-hoc clustering algorithm that works in $\widetilde{O}(n)$ time. The algorithm is based on random sampling to compute approximate centres for the clusters and on approximate nearest-neighbour data structures [30] to cluster the points around those centres.

Throughout this chapter we assume $k = \omega(\log n)$, otherwise we already know how to implement Spectral Clustering in $\widetilde{O}(m)$ time by Corollary 4.1. We also need a slightly stronger assumption on $\Upsilon_G(k)$:

$$\Upsilon_G(k) \geq C \cdot k^5 \log^c k$$

for some large enough constant $C, c > 1$. The results in this chapter are summarised by the following theorem, which will be proved in Section 5.2.

Theorem 5.1. *Let $G = (V, E)$ be a graph of n vertices and m edges, and $k = \omega(\log n)$ be the number of clusters. Assume that $\Upsilon_G(k) = \lambda_{k+1}/\rho(k) \geq C \cdot k^5 \log^c k$ for some large enough constant $C, c > 1$, and $\{S_i\}_{i=1}^k$ is a k-way partition such that $\phi_G(S_i) \leq \rho(k)$. Then, there exists an algorithm which runs in $\widetilde{O}(m)$ time and outputs a k-way partition $\{A_i\}_{i=1}^k$ such that, for all $1 \leq i \leq k$,*

1. $\mathrm{vol}(A_i \triangle S_i) = \widetilde{O}\left(k^3/\Upsilon_G(k)\right) \mathrm{vol}(S_i)$
2. $\phi_G(A_i) \leq 1.1 \cdot \phi_G(S_i) + \widetilde{O}\left(k^3/\Upsilon_G(k)\right)$.

5.1 k-means clustering on the spectral embedding

In this section we show how to exploit the properties of the spectral embedding to design an ad-hoc k-means algorithm that work in $\widetilde{O}(n)$ time, independently of k. We assume we have at disposal an embedding $x : V \to \mathbb{R}^d$ in $d = O(\mathrm{poly}\log n)$ dimensions that approximates the spectral embedding $\{F(u)\}_{u \in V}$ defined in (4.1). More precisely, we assume that, for any $u, v \in V$ and a small enough $\varepsilon > 0$, $\{x(u)\}_{u \in V}$ satisfies the following two conditions.

$$\begin{aligned} \left(1 - \frac{1}{10 \log n}\right) \cdot \|F(u)\|^2 \leq \|x(u)\|^2 \leq \|F(u)\|^2, \\ \left(1 - \frac{1}{10 \log n}\right) \cdot \|F(u) - F(v)\|^2 \leq \|x(u) - x(v)\|^2 \leq \|F(u) - F(v)\|^2. \end{aligned} \tag{5.1}$$

The first condition states that u is mapped to a vector of approximately the same norm as $F(u)$, while the second states that the pairwise distances between points in $\{x(u)\}_{u \in V}$ are approximately the same as the distances between the corresponding points in $\{F(u)\}_{u \in V}$. Notice that the spectral embedding $\{F(u)\}_{u \in V}$ trivially satisfies (5.1). When $k = \omega(\log^c n)$ for any constant $c > 0$, however, $\{F(u)\}_{u \in V}$ doesn't satisfy

the requirement on the dimension of the embedding. In the next section we will show it is always possible to compute an embedding in $d = O(\operatorname{poly}\log n)$ dimensions satisfying (5.1) in $\widetilde{O}(m)$ time. In this section we just assume we have such embedding and design an algorithm that runs in $\widetilde{O}(m)$ time and achieves the same guarantees as Corollary 4.1.

The algorithm consists of two steps. In the *seeding* step, we compute k vertices $c_1, \ldots, c_k$ such that, with constant probability, $x(c_1), \ldots, x(c_k)$ are close to the points $p^{(1)}, \ldots, p^{(k)}$ defined in (4.3). In the *grouping* step, we simply assign each point $\{x(u)\}_{u \in V}$ to the closest candidate centre $x(c_i)$. Naively, this would take $\Omega(nk)$ time, but we show it can be implemented in $\widetilde{O}(n)$ time with approximate nearest-neighbour data structures.

5.1.1 The seeding step

To explain how the first step of the algorithm works, notice that by Lemma 4.3 $\left\|p^{(i)}\right\|^2$ is close to $1/\operatorname{vol}(S_i)$ for any $i = 1, \ldots, k$. Moreover, by Lemma 4.1, most of the points in a cluster S_i are concentrated around $p^{(i)}$. This means that sampling a vertex u with probability proportional to $d_u \|F(u)\|^2$ or, equivalently, $d_u \|x(u)\|^2$, the probability of sampling a vertex belonging to a specific cluster S_i is close to $1/k$. Therefore, sampling $\Theta(k \log k)$ vertices in this way ensures that, with constant probability, there is at least one vertex sampled from each cluster. We remark that we cannot achieve the same result by sampling $\Theta(k \log k)$ vertices uniformly at random. Consider for example a graph with a cluster of volume $O(\sqrt{n})$: we would need to sample at least $\Omega(\sqrt{n})$ vertices to hit that cluster with constant probability.

Since by Lemma 4.1 most vertices are mapped by the spectral embedding F very close to the corresponding point $p^{(i)}$, it is likely that all the $\Theta(k \log k)$ sampled vertices are mapped by F and x to points close to the centre of their corresponding cluster. To obtain *exactly* k points close to the centres of the k different clusters, we just need to "trim" redundant points, i.e., points that are too close to each other to belong to different clusters. We call this procedure SeedAndTrim and its formal description is given in Procedure 2.

We now analyse Procedure 2. For any $1 \leq i \leq k$, we define $\mathcal{E}_i$ to be the sum of the ℓ_2^2-distance between the points $\{F(u) : u \in S_i\}$ and $p^{(i)}$, i.e.,

$$\mathcal{E}_i \triangleq \sum_{u \in S_i} d_u \left\|F(u) - p^{(i)}\right\|^2. \tag{5.2}$$

Procedure 2 SeedAndTrim($\{x(u)\}_{u\in V}, k$)

input: an embedding $\{x(u)\}_{u\in V}$ and the number of clusters k
$K \triangleq 10k\log k$
for $i = 1, \ldots, K$ **do**
    set $c_i = u$ with probability proportional to $d_u\|x(u)\|^2$
for $i = 2, \ldots, K$ **do**
    delete all c_j with $j < i$ such that $\|x(c_i) - x(c_j)\|^2 < \|x(c_i)\|^2/20$
return the remaining sampled vertices

Given any parameter $\rho > 0$, we define the radius of S_i with respect to ρ as

$$R_i^\rho \triangleq \frac{\rho \cdot \mathcal{E}_i}{\mathrm{vol}(S_i)}$$

and define $\mathsf{CORE}_i^\rho \subseteq S_i$ by

$$\mathsf{CORE}_i^\rho \triangleq \left\{u \in S_i : \left\|F(u) - p^{(i)}\right\|^2 \leq R_i^\rho\right\}. \tag{5.3}$$

We will show that, for a certain value of ρ, with constant probability Procedure 2 outputs k points, each one belonging to a different CORE_i^ρ. Before proceeding with the analysis of Procedure 2, a clarification is in order. We want to study the distribution of the sampled points with respect to the embedding x. The points $p^{(i)}$'s, however, are defined with respect to the spectral embedding F. Hence, we first need to understand where the points are mapped by F and then reason about their images with respect to x by taking advantage of conditions (5.1). Furthermore, notice that CORE_i^ρ represents the set of vertices in S_i that are mapped by F to points in the ball of (squared) radius R_i^ρ around $p^{(i)}$. From our discussion in Section 4.2, we know that larger clusters enjoy a higher degree of concentration than smaller ones, and that's why the radius R_i^ρ is inversely proportional to $\mathrm{vol}(S_i)$.

We now bound the volume of the points outside the core of a cluster. By the definition of CORE_i^ρ and an averaging argument, it holds that

$$\mathrm{vol}(S_i \setminus \mathsf{CORE}_i^\rho) \leq \frac{\sum_{u\in S_i} d_u \left\|F(u) - p^{(i)}\right\|^2}{R_i^\rho} = \frac{\mathrm{vol}(S_i)}{\rho}.$$

Therefore, for any $\rho \geq 0$, we have that

$$\mathrm{vol}(\mathsf{CORE}_i^\rho) \geq \max\left\{\left(1 - \frac{1}{\rho}\right)\mathrm{vol}(S_i), 0\right\}. \tag{5.4}$$

We set the parameter

$$\alpha \triangleq 100K \log K,$$

where $K = 10k \log k$ was defined in Procedure 2, and from now on we study CORE_i^α for $1 \le i \le k$. The next lemma shows that the probability mass is mostly concentrated on the vertices in the cores.

Lemma 5.1. *The following statements hold:*

1. $\sum_{u \in \mathsf{CORE}_i^\alpha} d_u \cdot \|F(u)\|^2 \ge 1 - \frac{1}{100K}$.
2. $\sum_{i=1}^k \sum_{u \notin \mathsf{CORE}_i^\alpha} d_u \cdot \|F(u)\|^2 \le \frac{k}{100K}$.

Proof. By the definition of CORE_i^α, it holds that

$$\begin{aligned}
&\sum_{u \in \mathsf{CORE}_i^\alpha} d_u \cdot \|F(u)\|^2 \\
&\ge \frac{1}{\alpha} \int_0^\alpha \sum_{u \in \mathsf{CORE}_i^\rho} d_u \cdot \|F(u)\|^2 d\rho \\
&\ge \frac{1}{\alpha} \int_0^\alpha \left(\left\| p^{(i)} \right\| - \sqrt{R_i^\rho} \right)^2 \mathrm{vol}(\mathsf{CORE}_i^\rho) d\rho && (5.5) \\
&\ge \frac{1}{\alpha} \int_0^\alpha \left(\left\| p^{(i)} \right\|^2 - 2\sqrt{R_i^\rho} \cdot \left\| p^{(i)} \right\| \right) \max\left\{ \left(1 - \frac{1}{\rho}\right) \mathrm{vol}(S_i), 0 \right\} d\rho && (5.6) \\
&\ge \frac{1}{\alpha} \int_0^\alpha \max\left\{ \left(1 - \frac{7k}{\sqrt{\Upsilon_G(k)}} - 3\sqrt{\mathcal{E}_i \rho} \right) \left(1 - \frac{1}{\rho}\right), 0 \right\} d\rho && (5.7)
\end{aligned}$$

where (5.5) follows from the fact that for all $u \in \mathsf{CORE}_i^\rho$, $\|F(u)\| \ge \|p^{(i)}\| - \sqrt{R_i^\rho}$, (5.6) from (5.4), and (5.7) from the definition of R_i^ρ and the fact (Lemma 4.3) that $1 - 7k \cdot (\Upsilon_G(k))^{-1/2} \le \|p^{(i)}\|^2 \cdot \mathrm{vol}(S_i) \le 1 + 7k \cdot (\Upsilon_G(k))^{-1/2}$. Since $\mathcal{E}_i \le 1.1k^2/\Upsilon_G(k)$ by Lemma 4.1, it holds that

$$\begin{aligned}
\sum_{u \in \mathsf{CORE}_i^\alpha} d_u \cdot \|F(u)\|^2 &\ge \frac{1}{\alpha} \int_0^\alpha \max\left\{ \left(1 - \frac{7k}{\sqrt{\Upsilon_G(k)}} - 4\sqrt{k^2\rho/\Upsilon_G(k)} \right) \left(1 - \frac{1}{\rho}\right), 0 \right\} d\rho \\
&\ge \frac{1}{\alpha} \int_0^\alpha \max\left\{ 1 - \frac{7k}{\sqrt{\Upsilon_G(k)}} - 4\sqrt{k^2\rho/\Upsilon_G(k)} - \frac{1}{\rho}, 0 \right\} d\rho \\
&\ge 1 - \frac{7k}{\sqrt{\Upsilon_G(k)}} - 4k\sqrt{\alpha/\Upsilon_G(k)} - \frac{\ln \alpha}{\alpha} \\
&\ge 1 - \frac{1}{100K},
\end{aligned}$$

where the last inequality holds by the assumption on α and $\Upsilon_G(k)$.

The second statement follows by the fact that

$$\sum_{i=1}^{k} \sum_{u \in \mathsf{CORE}_i^\alpha} d_u \cdot \|F(u)\|^2 \geq k \left(1 - \frac{1}{100K}\right)$$

and

$$\sum_{u \in V} d_u \cdot \|F(u)\|^2 = \sum_{u \in V} \| \left(f_1(u), \ldots, f_k(u)\right)^\intercal \|^2 = \sum_{i=1}^{k} \sum_{u \in V} f_i(u)^2 = k.$$

□

The next lemma shows that vertices from the same core are mapped by x to points close to each other, while vertices from different cores are mapped far away from each other.

Lemma 5.2. *The following statements hold:*

1. *For any $1 \leq i \leq k$ and any two vertices $u, v \in \mathsf{CORE}_i^\alpha$, it holds that*

$$\|x(u) - x(v)\|^2 \leq \min\left\{\frac{5\alpha k^2}{\Upsilon \operatorname{vol}(S_i)}, \frac{\|x(u)\|^2}{20k}\right\}.$$

2. *For any $i \neq j$, and $u \in \mathsf{CORE}_i^\alpha, v \in \mathsf{CORE}_j^\alpha$, it holds that*

$$\|x(u) - x(v)\|^2 \geq \frac{1}{4 \operatorname{vol}(S_i)} > \frac{\|x(u)\|^2}{10}.$$

Proof. By the definition of CORE_i^α, it holds for any $u \in \mathsf{CORE}_i^\alpha$ that $\left\|F(u) - p^{(i)}\right\| \leq \sqrt{R_i^\alpha}$. By the triangle inequality, it holds for any $u, v \in \mathsf{CORE}_i^\alpha$ that $\|F(u) - F(v)\| \leq 2\sqrt{R_i^\alpha}$. Hence,

$$\|F(u) - F(v)\|^2 \leq 4R_i^\alpha = \frac{4\alpha \mathcal{E}_i}{\operatorname{vol}(S_i)} \leq \frac{5\alpha k^2}{\Upsilon \operatorname{vol}(S_i)},$$

where the last inequality follows from Lemma 4.1. Hence, by the assumption (5.1) on the embedding x, it holds that

$$\|x(u) - x(v)\|^2 \leq \|F(u) - F(v)\|^2 \leq \frac{5\alpha k^2}{\Upsilon \operatorname{vol}(S_i)}.$$

On the other hand, we have that

$$\|F(u)\|^2 \geq \left(\left\|p^{(i)}\right\| - \sqrt{R_i^\alpha}\right)^2 \geq \frac{9}{10\,\mathrm{vol}(S_i)},$$

where the last inequality follows from Lemma 4.3 and the definition of R_i^α. By (5.1) and the conditions on α and Υ, it also holds that

$$\|x(u) - x(v)\|^2 \leq \frac{5\alpha k^2}{\Upsilon\,\mathrm{vol}(S_i)} \leq \frac{6\alpha k^2}{\Upsilon}\|F(u)\|^2 \leq \frac{\|x(u)\|^2}{20k}.$$

With these we proved the first statement.

Now for the second statement. By the triangle inequality, it holds for any pair of $u \in \mathsf{CORE}_i^\alpha$ and $v \in \mathsf{CORE}_j^\alpha$ that

$$\|F(u) - F(v)\| \geq \left\|p^{(i)} - p^{(j)}\right\| - \left\|F(u) - p^{(i)}\right\| - \left\|F(v) - p^{(j)}\right\|.$$

By Lemma 4.4, we have for any $i \neq j$ that

$$\left\|p^{(i)} - p^{(j)}\right\|^2 \geq \frac{1}{2\min\left\{\mathrm{vol}(S_i), \mathrm{vol}(S_j)\right\}}.$$

Combining this with the fact that

$$\left\|F(u) - p^{(i)}\right\| \leq \sqrt{R_i^\alpha} \leq \sqrt{\frac{1.1\alpha k^2}{\Upsilon\,\mathrm{vol}(S_i)}},$$

we obtain that

$$\begin{aligned}
\|F(u) - F(v)\| &\geq \left\|p^{(i)} - p^{(j)}\right\| - \left\|F(u) - p^{(i)}\right\| - \left\|F(v) - p^{(j)}\right\| \\
&\geq \sqrt{\frac{1}{2\min\left\{\mathrm{vol}(S_i), \mathrm{vol}(S_j)\right\}}} - \sqrt{\frac{1.1\alpha k^2}{\Upsilon\,\mathrm{vol}(S_i)}} - \sqrt{\frac{1.1\alpha k^2}{\Upsilon\,\mathrm{vol}(S_j)}} \\
&\geq \sqrt{\frac{1}{3\min\left\{\mathrm{vol}(S_i), \mathrm{vol}(S_j)\right\}}}.
\end{aligned}$$

Notice that $\|x(u)\|^2 \leq \|F(u)\|^2 \leq \left(\left\|p^{(i)}\right\| + \sqrt{R_i^\alpha}\right)^2 \leq \frac{11}{10\,\mathrm{vol}(S_i)}$, therefore we have

$$\|x(u) - x(v)\|^2 \geq \left(1 - \frac{1}{10\log n}\right)\|F(u) - F(v)\|^2 \geq \frac{1}{4\,\mathrm{vol}(S_i)} > \frac{\|x(u)\|^2}{10}. \qquad \square$$

We now show that if we sample $\Theta(k \log k)$ vertices, with constant probability they will all belong to the cores of the clusters, and each core will contain at least one sampled vertex.

Lemma 5.3. *Suppose we sample $K = 10k \log k$ vertices, and each vertex $u \in V$ is sampled with probability proportional to $d_u \cdot \|x(u)\|^2$. Then, with constant probability, the set $C = \{c_1 \dots c_K\}$ of sampled vertices satisfies the following properties:*

1. *C only contains vertices from the cores, i.e., $C \subseteq \bigcup_{i=1}^{k} \mathsf{CORE}_i^{\alpha}$;*
2. *C contains at least one vertex from each cluster, i.e., $C \cap S_i \neq \emptyset$ for any $1 \leq i \leq k$.*

Proof. By (5.1), it holds for every vertex u that

$$\left(1 - \frac{1}{10 \log n}\right) \cdot \|F(u)\|^2 \leq \|x(u)\|^2 \leq \|F(u)\|^2.$$

Since $\sum_{u \in V} d_u \|F(u)\|^2 = k$, it holds that $\sum_{u \in V} d_u \|x(u)\|^2 \leq \sum_{u \in V} d_u \|F(u)\|^2 = k$ and

$$\sum_{u \in V} d_u \|x(u)\|^2 \geq \sum_{u \in V} d_u \cdot \left(1 - \frac{1}{10 \log n}\right) \cdot \|F(u)\|^2 = \left(1 - \frac{1}{10 \log n}\right) \cdot k.$$

Hence, the total probability mass that we use to sample vertices, i.e., $\sum_{u \in V} d_u \|x(u)\|^2$, is between $\left(1 - \frac{1}{10 \log n}\right) \cdot k$ and k.

We first bound the probability that we sample at least one vertex from each core. By Lemma 5.1 we have that $\sum_{u \in \mathsf{CORE}_i^{\alpha}} d_u \cdot \|F(u)\|^2 \geq 1 - \frac{1}{100K}$. Hence, for any fixed $1 \leq i \leq k$, the probability that a vertex from CORE_i^{α} gets sampled is at least

$$\frac{\sum_{u \in \mathsf{CORE}_i^{\alpha}} d_u \cdot \|x(u)\|^2}{k} \geq \frac{\sum_{u \in \mathsf{CORE}_i^{\alpha}} d_u \cdot \|F(u)\|^2}{2k} \geq \frac{1 - \frac{1}{100K}}{2k} \geq \frac{1}{3k}.$$

Therefore, the probability that we never encounter a vertex from CORE_i^{α} after sampling K vertices is at most $\left(1 - \frac{1}{3k}\right)^K \leq \frac{1}{10k}$. Since by Lemma 5.1 we have that $\sum_{i=1}^{k} \sum_{u \notin \mathsf{CORE}_i^{\alpha}} d_u \cdot \|F(u)\|^2 \leq \frac{k}{100K}$, the probability that a sampled vertex is outside the cores of the clusters is at most

$$\begin{aligned}\frac{\sum_{i=1}^{k} \sum_{u \in S_i \setminus \mathsf{CORE}_i^{\alpha}} d_u \cdot \|x(u)\|^2}{\left(1 - \frac{1}{10 \log n}\right) \cdot k} &\leq \frac{\sum_{i=1}^{k} \sum_{u \in S_i \setminus \mathsf{CORE}_i^{\alpha}} d_u \|F(u)\|^2}{k/2} \\ &\leq \frac{k}{100K} \cdot \frac{2}{k} = \frac{1}{50K}.\end{aligned}$$

Taking a union bound over all the undesired events, the probability that there exists at least one sampled vertex in $V \setminus \left(\bigcap_{i=1}^{k} \mathsf{CORE}_i^{\alpha}\right)$ or that there exists $1 \leq i \leq k$ such that $C \cap \mathsf{CORE}_i^{\alpha} = \emptyset$ is at most $k \cdot \frac{1}{10k} + K \cdot \frac{1}{50K} \leq \frac{1}{5}$. □

Based on Lemma 5.2 and Lemma 5.3, to delete vertices belonging to the same core we can simply delete any one of two vertices c_i and c_j whose distance is less than $\|x(c_i)\|^2/20$. The following lemma presents the correctness and runtime of the procedure SeedAndTrim, i.e., Procedure 2.

Lemma 5.4. *Given the embedding $\{x(u)\}_{u\in V}$ of dimension $d = O(\operatorname{poly}\log n)$ that satisfies* (5.1)*, with constant probability* SeedAndTrim *returns a set $C^\star$ of centres $c_1 \ldots c_k$ in $\widetilde{O}(n + k^2)$ time, such that each CORE_i^{α} contains exactly one vertex in $C^\star$.*

Proof. By Lemma 5.3, the sampled set C contains at least one vertex from each core CORE_i^{α} with constant probability, while Lemma 5.2 and the algorithm description ensure that only vertices from different cores will remain in $C^\star$. Therefore, SeedAndTrim returns exactly k points from different cores with constant probability.

Now we analyse the runtime. The procedure takes $\widetilde{O}(n)$ time to compute the norms of $\{x(u)\}_{u\in V}$, since the embedding has dimension $O(\operatorname{poly}\log n)$ by assumption. It takes $\widetilde{O}(k)$ time to sample $\widetilde{O}(k)$ vertices, and trimming the sampling vertices takes at most $\widetilde{O}(k^2)$ time. Hence, the total runtime is $\widetilde{O}(n + k^2)$. □

As the end of this section, we would like to mention that choosing good candidate centres is crucial for most k-means algorithms, and has been studied extensively in the literature, e.g., [9, 52]. Comparing with recent algorithms that obtain good initial centres by iteratively picking points from a *non-uniform* distribution and take $\Omega(nk)$ time, our seeding step (Algorithm 2) runs in $\widetilde{O}(n + k^2)$ time.

5.1.2 The grouping step

At the beginning of the second step we assume we have a set of k candidate centres $C^\star = \{c_1, \ldots, c_k\}$ such that each c_i is in the core of the corresponding cluster S_i. The goal of the grouping step is to assign every vertex $u \in V$ to one of the candidate centre. In particular, we want to assign u to a centre $c_i \in C^\star$ minimising the ℓ_2-distance between $x(u)$ and $x(c_i)$. Notice that a naive implementation of this step requires $\Omega(nk)$ time: for each vertex u we would need to compute the distance between $x(u)$ and $x(c_i)$ for any $1 \leq i \leq k$. Since in our case most of the points are close to just one of the candidate centres and far from all the others, we don't need to compute all these

distances precisely. Instead, we can just repeatedly solve an ε-approximate nearest neighbour problem (ε-NNS) [30]:

Problem 5.1 (ε-approximate nearest neighbour problem)**.** *Given a set of points $P \subset \mathbb{R}^d$ and a point $q \in \mathbb{R}^d$, find a point $p \in P$ such that, for all $p' \in P$, $\|p - q\| \leq (1+\varepsilon)\|p' - q\|$.*

An efficient data structure to solve this problem was given in [30]:

Theorem 5.2 ([30])**.** *Given a set P of points in $\mathbb{R}^d$, there is an algorithm that solves the ε-approximate nearest neighbour problem with $\widetilde{O}\left(|P|^{1+\frac{1}{1+\varepsilon}} + d \cdot |P|\right)$ preprocessing time and $\widetilde{O}\left(d \cdot |P|^{\frac{1}{1+\varepsilon}}\right)$ query time.*

Our grouping step works as follows. We set $P = \{x(c_1), \ldots, x(c_k)\}$, $\varepsilon = \log k - 1$, and apply the above ε-approximate nearest neighbour data structures to assign all the points in our embedding $\{x(u)\}_{u \in V}$ to their (approximately) nearest candidate centre in the set P. We group all the vertices assigned to the centre $x(c_i)$ in a single cluster A_i and we return the partition $A_1, \ldots, A_k$. By Theorem 5.2, this procedure requires only $\widetilde{O}(k)$ preprocessing time and $\widetilde{O}(1)$ time for each query. Hence, the runtime of the grouping step is $\widetilde{O}(n)$. Notice that, with our choice of $\varepsilon = \log k - 1$ and application of ε-NNS, all the remaining vertices in $V \setminus C^\star$ *may not* be assigned to the cluster A_i with the nearest centre c_i. We will prove shortly that our choice of ε suffices to obtain a good approximation of the optimal partition. The runtime of the grouping step and the properties of the returned clusters are summarised in the following lemma:

Lemma 5.5. *Given a set of centres $C^\star = \{c_1, \ldots, c_k\}$, the grouping step runs in $\widetilde{O}(n)$ time and returns a partition $A_1, \ldots, A_k$ of vertices such that, for any $1 \leq i \neq j \leq k$ and every $u \in A_i$, it holds that $\|x(u) - x(c_i)\| \leq \log k \cdot \|x(u) - x(c_j)\|$.*

5.1.3 Approximation analysis

We now give an analysis of the approximation guarantees for the partition $A_1, \ldots, A_k \subset V$ output by the grouping step. The next lemma bounds the symmetric difference between $\{A_i\}_{i=1}^k$ and the optimal partition $\{S_i\}_{i=1}^k$.

Lemma 5.6. *Let $A_1, \ldots, A_k$ be the output of the grouping procedure. Then, under a proper permutation of the indices, with constant probability, for any $1 \leq i \leq k$ it holds that (i) $\operatorname{vol}(A_i \triangle S_i) = \widetilde{O}\left(k^2/\Upsilon\right) \operatorname{vol}(S_i)$, and (ii) $\phi_G(A_i) \leq 1.1 \cdot \phi_G(S_i) + \widetilde{O}\left(k^2/\Upsilon\right)$.*

Proof. We assume that $c_1, \dots, c_k \in V$ are the centres returned by SeedAndTrim, and $\{x(u)\}_{u \in V}$ is the embedding we use in the algorithm. Moreover, $\{x(u)\}_{u \in V}$ satisfies (5.1). We further assume that $\{c_1, \dots, c_k\} \subseteq \bigcup_{i=1}^{k} \mathsf{CORE}_i^{\alpha}$. By Lemma 5.3, this holds with constant probability, and we assume that this event happens in the following analysis. Then, by the second statement of Lemma 5.2 it holds for any $i \neq j$ that

$$\|x(c_i) - x(c_j)\|^2 = \Omega\left(\frac{1}{\min\{\mathrm{vol}(S_i), \mathrm{vol}(S_j)\}}\right). \tag{5.8}$$

By Lemma 5.5, it holds for any $1 \leq i \leq k$ that

$$\begin{aligned}
&\mathrm{vol}(S_i \setminus A_i) \\
&\leq \sum_{i \neq j} \mathrm{vol}\left(\left\{v \in S_i : \|x(c_i) - x(v)\| > \frac{\|x(c_j) - x(v)\|}{\log k}\right\}\right) \\
&\leq \sum_{i \neq j} \mathrm{vol}\left(\left\{v \in S_i : \|x(c_i) - x(v)\| > \frac{\|x(c_i) - x(c_j)\| - \|x(c_i) - x(v)\|}{\log k}\right\}\right) \\
&\leq \sum_{i \neq j} \mathrm{vol}\left(\left\{v \in S_i : 2\|x(c_i) - x(v)\| > \frac{\|x(c_i) - x(c_j)\|}{\log k}\right\}\right) \\
&= \sum_{i \neq j} \mathrm{vol}\left(\left\{v \in S_i : \|x(c_i) - x(v)\| > \frac{\|x(c_i) - x(c_j)\|}{2\log k}\right\}\right).
\end{aligned}$$

By (5.1) and the triangle inequality, we have that $\|x(c_i) - x(v)\| \leq \|F(c_i) - F(v)\| \leq \left\|F(c_i) - p^{(i)}\right\| + \left\|p^{(i)} - F(v)\right\|$. Therefore,

$$\begin{aligned}
&\mathrm{vol}(S_i \setminus A_i) \\
&\leq \sum_{i \neq j} \mathrm{vol}\left(\left\{v \in S_i : \left\|F(c_i) - p^{(i)}\right\| + \left\|p^{(i)} - F(v)\right\| > \frac{\|x(c_i) - x(c_j)\|}{2\log k}\right\}\right) \\
&\leq \sum_{i \neq j} \mathrm{vol}\left(\left\{v \in S_i : \left\|p^{(i)} - F(v)\right\| > \frac{\|x(c_i) - x(c_j)\|}{2\log k} - \left\|F(c_i) - p^{(i)}\right\|\right\}\right) \\
&\leq \sum_{i \neq j} \mathrm{vol}\left(\left\{v \in S_i : \left\|p^{(i)} - F(v)\right\| > \frac{\|x(c_i) - x(c_j)\|}{2\log k} - \sqrt{R_i^{\alpha}}\right\}\right) \\
&\leq \sum_{i \neq j} \mathrm{vol}\left(\left\{v \in S_i : \left\|p^{(i)} - F(v)\right\|^2 = \Omega\left(\frac{1}{\log^2 k \cdot \min\{\mathrm{vol}(S_i), \mathrm{vol}(S_j)\}}\right)\right\}\right) \\
&= \widetilde{O}\left(k^2/\Upsilon_G(k)\right)\mathrm{vol}(S_i),
\end{aligned}$$

where the last equality follows from Lemma 4.1. By the same argument, it also follows that

$$\begin{aligned}\operatorname{vol}(A_i \setminus S_i) &\leq \sum_{i \neq j} \operatorname{vol}\left(\left\{v \in S_j : \|x(c_j) - x(v)\| \geq \frac{\|x(c_i) - x(v)\|}{\log k}\right\}\right) \\ &= \widetilde{O}\left(k^2/\Upsilon_G(k)\right) \operatorname{vol}(S_i),\end{aligned}$$

and therefore

$$\operatorname{vol}(S_i \triangle A_i) = \operatorname{vol}(S_i \setminus A_i) + \operatorname{vol}(A_i \setminus S_i) = \widetilde{O}\left(k^2/\Upsilon_G(k)\right) \operatorname{vol}(S_i).$$

This yields the first statement of the lemma.

The second statement follows by the same argument used in proving Theorem 4.1. □

5.2 Fast approximation of the spectral embedding

In Section 5.1 we presented an algorithm to partition a well-clustered graph that works in $\widetilde{O}(n)$ time, assuming one has at disposal an embedding $x : V \to \mathbb{R}^d$ with $d = O(\operatorname{poly}\log n)$ that satisfies (5.1). We now show how to compute such an embedding in $\widetilde{O}(m)$ time. We already argued that the spectral embedding (4.1) satisfies these condition when $k = O(\operatorname{poly}\log n)$ and, therefore, we need to consider only the case when k is large. For this reason, we will assume for the rest of this section that $k = \omega(\log n)$. Instead of the spectral embedding, we will work with the *heat kernel embedding* of the graph.

Formally, the heat kernel of G with parameter $t \geq 0$, called *temperature*, is defined as

$$H_t \triangleq \mathrm{e}^{-t\mathcal{L}_G} = \sum_{i=1}^{n} \mathrm{e}^{-t\lambda_i} f_i f_i^{\mathsf{T}}. \tag{5.9}$$

We view the heat kernel as a geometric embedding from V to $\mathbb{R}^n$ defined as

$$x_t(u) \triangleq \frac{1}{\sqrt{d_u}} \cdot \left(\mathrm{e}^{-t\cdot\lambda_1} f_1(u), \cdots, \mathrm{e}^{-t\cdot\lambda_n} f_n(u)\right), \tag{5.10}$$

and define the ℓ_2^2-distance between the points $x_t(u)$ and $x_t(v)$ as

$$\eta_t(u,v) \triangleq \|x_t(u) - x_t(v)\|^2. \tag{5.11}$$

Notice that, when G is well-clustered, there is a large gap between λ_k and λ_{k+1}. Therefore, if we set $t \approx \log n/\lambda_{k+1}$, the contribution of the eigenvalues $\lambda_{k+1}, \ldots, \lambda_n$ towards (5.10) is negligible, while the contribution of the bottom k eigenvalues $\lambda_1, \ldots, \lambda_k$ is approximately the same. This means that the heat kernel embedding approximates the spectral embedding. To compute the heat kernel embedding we need to compute the matrix exponential of the Laplacian of G. For this reason, we use the algorithm for approximating the matrix exponential proposed in [51], whose performance is summarised in Theorem 5.3. Recall that any $n \times n$ real and symmetric matrix A is diagonally dominant (SDD), if $A_{ii} \geq \sum_{j \neq i} |A_{ij}|$ for any $i = 1, \ldots, n$. It is easy to see that the Laplacian matrix of an undirected graph is diagonally dominant.

Theorem 5.3 ([51])**.** *Given an $n \times n$ SDD matrix A with m_A nonzero entries, a vector v and a parameter $\delta > 0$, there is an algorithm that can compute a vector x such that $\|\mathrm{e}^{-A}v - x\| \leq \delta\|v\|$ in time $\widetilde{O}((m_A + n)\log(2 + \|A\|))$, where the $\widetilde{O}(\cdot)$ notation hides* $\operatorname{poly}\log n$ *and* $\operatorname{poly}\log(1/\delta)$ *factors.*

Notice that the heat kernel embedding is n-dimensional. To obtain a low-dimensional embedding we use Johnson-Lindestrauss random projections [3, 31]. We exploit random projection also to avoid computing the matrix exponential/vector product n times, following an idea first proposed in [63]. The following lemma shows that, when $k = \Omega(\log n)$ and $\Upsilon_G(k) = \widetilde{\Omega}(k^4)$, we can compute in $\widetilde{O}(m)$ time an embedding $\{x(u)\}_{u \in V}$ in $O(\log^3 n)$ dimensions that satisfies (5.1) with high probability.

Lemma 5.7. *Let $k = \Omega(\log n)$ and assume $\Upsilon_G(k) = \lambda_{k+1}/\rho(k) \geq C \cdot k^5 \log^c k$ for some large enough constant $C, c > 1$. Then, there exists an embedding $x : V \to \mathbb{R}^d$ in $d = O(\operatorname{poly}\log n)$ dimensions such that, with high probability, the conditions* (5.1) *are satisfied. Moreover, if we know the values of λ_k and λ_{k+1}, this embedding can be computed in $\widetilde{O}(m)$ time.*

Proof. By the higher-order Cheeger inequality (2.10), we have that

$$\Upsilon = \frac{\lambda_{k+1}}{\rho(k)} \leq \frac{2\lambda_{k+1}}{\lambda_k}.$$

Since $k = \Omega(\log n)$ and $\Upsilon = \widetilde{\Omega}(k^4)$, it holds that $400 \cdot \log^2 n \leq \lambda_{k+1}/\lambda_k$, and there is a t such that

$$t \in \left(\frac{10 \cdot \log n}{\lambda_{k+1}}, \quad \frac{1}{20 \cdot \lambda_k \cdot \log n} \right). \tag{5.12}$$

We first show that the embedding $\{x_t(u)\}_{u \in V}$ with this t satisfies (5.1).

By the definition of $\eta_t(u,v)$, we have that

$$\begin{aligned}\eta_t(u,v) &= \sum_{i=1}^{n} \mathrm{e}^{-2t\lambda_i}\left(\frac{f_i(u)}{\sqrt{d_u}} - \frac{f_i(v)}{\sqrt{d_v}}\right)^2 \\ &= \sum_{i=1}^{k} \mathrm{e}^{-2t\lambda_i}\left(\frac{f_i(u)}{\sqrt{d_u}} - \frac{f_i(v)}{\sqrt{d_v}}\right)^2 + \sum_{i=k+1}^{n} \mathrm{e}^{-2t\lambda_i}\left(\frac{f_i(u)}{\sqrt{d_u}} - \frac{f_i(v)}{\sqrt{d_v}}\right)^2. \end{aligned} \tag{5.13}$$

Notice that it holds for $1 \leq i \leq k$ that

$$1 - \frac{1}{20\log n} \leq \mathrm{e}^{-1/(20\log n)} \leq \mathrm{e}^{-2t\lambda_i} \leq 1, \tag{5.14}$$

and it holds for $k+1 \leq i \leq n$ that

$$\mathrm{e}^{-2t\cdot\lambda_i} \leq \mathrm{e}^{-2\lambda_i\cdot 10\log n/\lambda_{k+1}} \leq \mathrm{e}^{-20\log n\lambda_{k+1}/\lambda_{k+1}} = \frac{1}{n^{20}}. \tag{5.15}$$

Combining (5.13), (5.14), and (5.15), it holds for any pair $u, v \in V$ that

$$\left(1 - \frac{1}{20\cdot\log n}\right)\cdot\|F(u) - F(v)\|^2 \leq \eta_t(u,v) \leq \|F(u) - F(v)\|^2 + \frac{1}{n^5}.$$

We can easily remove the $1/n^{10}$ factor, which comes from (5.15), by normalising the point in the embedding $\{x_t(u)\}_{u\in V}$ by a factor slightly smaller than one. Therefore, the embedding $\{x_t(u)\}_{u\in V}$ satisfies conditions (5.1)

Now we show that the distances $\|x_t(u) - x_t(v)\|$ for all pairs $u, v \in V$ can be approximately computed in nearly-linear time. For any vertex $u \in V$, we define $\xi_u \in \mathbb{R}^n$, where $(\xi_u)_v = 1/\sqrt{d_u}$ if $v = u$, and $(\xi_u)_v = 0$ otherwise. Combining (5.9) with (5.10) and (5.11), we have that $\eta_t(u,v) = \|H_t\,(\xi_u - \xi_v)\|^2$. We define Z to be the operator of error δ which corresponds to the algorithm described in Theorem 5.3, and replacing H_t with Z we get

$$\left|\|Z\,(\xi_u - \xi_v)\| - \eta_t^{1/2}(u,v)\right| \leq \delta\,\|\xi_u - \xi_v\| \leq \delta,$$

where the last inequality follows from $d_u, d_v \geq 1$. Hence, it holds that

$$\eta_t^{1/2}(u,v) - \delta \leq \|Z\,(\xi_u - \xi_v)\| \leq \eta_t^{1/2}(u,v) + \delta. \tag{5.16}$$

We apply the Johnson-Lindenstrauss transform in a way analogous to the computation of effective resistances [63]: we construct an $O(\varepsilon^{-2}\cdot\log n)\times n$ Gaussian matrix Q such

that, with high probability, it holds for all $u, v \in V$ that

$$(1-\varepsilon)\left\|Z\left(\xi_u-\xi_v\right)\right\| \leq\left\|QZ\left(\xi_u-\xi_v\right)\right\| \leq(1+\varepsilon)\left\|Z\left(\xi_u-\xi_v\right)\right\|. \tag{5.17}$$

Combining (5.16) and (5.17) gives us that

$$(1-\varepsilon)\left(\eta_t^{1/2}(u, v)-\delta\right) \leq\left\|QZ\left(\xi_u-\xi_v\right)\right\| \leq(1+\varepsilon)\left(\eta_t^{1/2}(u, v)+\delta\right).$$

Squaring both sides and invoking the inequality $(1-\varepsilon)a^2-(1+\varepsilon^{-1})b^2 \leq (a+b)^2 \leq (1+\varepsilon)a^2+(1+\varepsilon^{-1})b^2$ gives

$$(1-5\varepsilon)\,\eta_t(u, v)-2\delta^2\varepsilon^{-1} \leq\left\|QZ\left(\xi_u-\xi_v\right)\right\|^2 \leq(1+5\varepsilon)\,\eta_t(u, v)+2\delta^2\varepsilon^{-1}$$

Scaling QZ by a factor of $(1+5\varepsilon)^{-1}$, and appending an extra entry in each vector to create an additive distortion of $2\delta\varepsilon^{-1}$ then gives the desired bounds when δ is set to εn^{-6}. To satisfy the conditions (5.1) we just need to set $\varepsilon = O(1/\log n)$.

To analyse the runtime of computing $QZ\xi_u$ for all $u \in V$, notice that Q has only $O(\log^3 n)$ rows. We can then run the algorithm for approximately computing the matrix exponential from [51] $O(\log^3 n)$ times, where each time we use a different row of Q as input, obtaining a row of QZ as output. Since $\|\mathcal{L}_G\| \leq 2$, by Theorem 5.3 we can compute QZ in $\widetilde{O}(m)$ time. Notice that $QZ\xi_u$ is just some column of QZ after rescaling, therefore we can compute all the required distances in time $\widetilde{O}(m)$. □

We remark that the proof above shows an interesting property about the embedding (5.10), i.e., for a large value of k and a certain condition on Υ, there is always a t such that the values of $\eta_t(u, v)$ gives a good approximation of $\|F(u)-F(v)\|^2$ for all pair of vertices u, v. Unfortunately, it is unclear how to compute a correct value of t without knowing λ_k, and computing λ_k seems to be equivalent to computing the spectral embedding, which was exactly what we were trying to avoid.

To overcome this obstacle, we can iterate for all possible $0 < t \leq n^{10}$ of the form 2^i, and for each value of t compute the embedding $\{x_t(u)\}_{u \in V}$ and run the seeding and grouping steps on this embedding. In this way, we are guaranteed that at some point we will hit a t such that $\eta_t(u, v)$ well-approximates $\|F(u)-F(v)\|^2$ and Lemma 5.7 holds. Unfortunately, it's not clear how to recognise when we hit the correct value of t. However, as t gets larger, the pairwise distances shrink and, as $t \to \infty$, the points in the embedding will tend to concentrate around a single point. For this reason, we can detect when t has become too large just looking at the total norm of the

points in the embedding. More precisely, for every possible t we compute the value of $\sum_{v\in V} d_v\|x_t(v)\|^2$, and the algorithm only moves to the next iteration if

$$\sum_{v\in V} d_v\|x_t(v)\|^2 \geq k\left(1-\frac{2}{\log n}\right). \tag{5.18}$$

By the proof of Lemma 5.7, (5.18) is satisfied for all values of t in the right range (5.12), and the algorithm will not terminate before $t=\lfloor \log n/\lambda_{k+1}\rfloor$. Moreover, if the pairwise distances $\eta_t(u,v)$ have become too small, we will prove that SeedAndTrim will return strictly less than k points. See Algorithm 3 for the formal description of our final algorithm.

Algorithm 3 A nearly-linear time graph clustering algorithm, $k=\Omega(\log n)$

1: **input**: the input graph G, and the number of clusters k
2: Let $t=2$.
3: **repeat**
4: Let $(c_1,\ldots,c_k)=\textsc{SeedAndTrim}(k,\{x_t(u)\}_{u\in V})$.
5: **if** SeedAndTrim returns exactly k points **then**
6: Compute a partition $A_1,\ldots,A_k$ of V: for every $v\in V$ assign v to its nearest centre c_i using the ε-NNS algorithm with $\varepsilon=\log k-1$.
7: Let $t=2t$
8: **until** $t>n^{10}$ **or** $\sum_{v\in V} d_v\|x_t\|^2 < k\left(1-\frac{2}{\log n}\right)$.
9: **return** $(A_1,\cdots,A_k)$.

Lemma 5.8. *Let $t=\Omega\left(1/(\lambda_k\cdot\log n)\right)$, and t satisfies (5.18). Suppose that Procedure 2 uses the embedding $\{x_t(u)\}_{u\in V}$ and returns k centres $c_1,\ldots,c_k$. Then, with constant probability, the following statements hold:*

1. *It holds that*
$$\{c_1,\ldots,c_k\}\subseteq\bigcup_{i=1}^{k}\mathsf{CORE}_i^{\alpha}.$$
2. *These k centres belong to different cores, and it holds for any different i,j that*
$$\|x_t(c_i)-x_t(c_j)\|^2=\widetilde{\Omega}\left(\frac{1}{\mathrm{vol}(S_i)}\right).$$
3. *For any $i=1,\ldots,k$, it holds that*
$$\sum_{i=1}^{k}\sum_{u\in S_i} d_u\cdot\|x(u)-x(c_i)\|^2=\widetilde{O}\left(\frac{k^2}{\Upsilon_G(k)}\right).$$

Proof. Since $\|x_t(u)\|$ is decreasing with respect to the value of t for any vertex u, by Lemma 5.1 for any $t = \Omega\left(1/(\lambda_k \cdot \log n)\right)$ we have:

$$\sum_{i=1}^{k} \sum_{u \notin \mathsf{CORE}_i^\alpha} d_u \cdot \|x_t(u)\|^2 \leq \sum_{i=1}^{k} \sum_{u \notin \mathsf{CORE}_i^\alpha} d_u \cdot \|F(u)\|^2 \leq \frac{k}{100K} \leq \frac{1}{\log k}.$$

On the other hand, we only consider values of t satisfying (5.18). Since every vertex u is sampled with probability proportional to $d_u \cdot \|x_t(u)\|^2$, with constant probability it holds that

$$\{c_1, \ldots, c_k\} \subseteq \bigcup_{i=1}^{k} \mathsf{CORE}_i^\alpha,$$

which proves the first statement.

Now we prove that these k centres belong to different cores. We fix an index i, and assume that $c_i \in S_i$. We will prove that

$$\|x_t(c_i)\|^2 = \widetilde{\Omega}\left(\frac{1}{\mathrm{vol}(S_i)}\right). \tag{5.19}$$

Assume by contradiction that (5.19) does not hold, i.e.,

$$\|x_t(c_i)\|^2 = o\left(\frac{1}{\log^c k \,\mathrm{vol}(S_i)}\right)$$

for any constant c. Then, we have that

$$\begin{aligned}
\sum_{u \in \mathsf{CORE}_i^\alpha} d_u \cdot \|x_t(u)\|^2 &\leq \sum_{u \in \mathsf{CORE}_i^\alpha} d_u \cdot \left(\|x_t(c_i)\| + \sqrt{R_i^\alpha}\right)^2 \\
&\leq 2 \cdot \sum_{u \in \mathsf{CORE}_i^\alpha} \left(d_u \cdot \|x_t(c_i)\|^2 + d_u \cdot R_i^\alpha\right) \\
&= o\left(\frac{1}{\log^c k}\right).
\end{aligned}$$

Combining this with (5.18), the probability that vertices get sampled from CORE_i^α is

$$\frac{\sum_{u \in \mathsf{CORE}_i^\alpha} d_u \cdot \|x_t(u)\|^2}{\sum_{v \in V} d_v \|x_t(v)\|^2} = o\left(\frac{1}{k \cdot \log^c k}\right).$$

This means if we sample $K = \Theta(k \log k)$ vertices, vertices in CORE_i^α will not get sampled with probability at least $1 - 1/\log^5 k$. This contradicts the fact that $c_i \in \mathsf{CORE}_i^\alpha$. Therefore (5.19) holds.

Now, by description of Procedure 2, we have for any $j \neq i$:

$$\|x_t(c_i) - x_t(c_j)\|^2 \geq \frac{\|x(c_i)\|^2}{20} = \widetilde{\Omega}\left(\frac{1}{\mathrm{vol}(S_i)}\right),$$

where the last equality follows from (5.19). Since any vertex in CORE_i^α has distance at most R_i^α from c_i, c_j and c_i belong to different cores. Therefore, the second statement holds.

Finally we turn our attention to the third statement. We showed in Lemma 5.7 that, when $t = \Theta\left(1/(\lambda_k \cdot \log n)\right)$, the embedding $\{x_t(u)\}_{u\in V}$ satisfies the conditions (5.1). Hence, it holds that

$$\begin{aligned}
&\sum_{i=1}^{k}\sum_{u\in S_i} d_u \cdot \|x(u) - x(c_i)\|^2 \\
&\leq \sum_{i=1}^{k}\sum_{u\in S_i} d_u \cdot \|F(u) - F(c_i)\|^2 \\
&\leq \sum_{i=1}^{k}\sum_{u\in S_i} d_u \cdot \left(\left\|F(u) - p^{(i)}\right\| + \left\|F(c_i) - p^{(i)}\right\|\right)^2 \\
&\leq \sum_{i=1}^{k}\sum_{u\in S_i} 2\cdot d_u \cdot \left(\left\|F(u) - p^{(i)}\right\|^2 + \left\|F(c_i) - p^{(i)}\right\|^2\right) \qquad (5.20)
\end{aligned}$$

Notice that by Lemma 4.1 we have

$$\sum_{i=1}^{k}\sum_{u\in S_i} d_u \cdot \|F(u) - p^{(i)}\|^2 \leq 1.1k^2/\Upsilon_G(k). \qquad (5.21)$$

On the other hand, we have $\|F(c_i) - p^{(i)}\|^2 \leq R_i^\alpha$ as $c_i \in \mathsf{CORE}_i^\alpha$, and

$$\sum_{i=1}^{k}\sum_{u\in S_i} 2\cdot d_u \cdot \left\|F(c_i) - p^{(i)}\right\|^2 \leq \sum_{i=1}^{k} 2\,\mathrm{vol}(S_i)\cdot\frac{\alpha\cdot\mathcal{E}_i}{\mathrm{vol}(S_i)} = \sum_{i=1}^{k} 2\alpha\cdot\mathcal{E}_i = \widetilde{O}\left(\frac{k^3}{\Upsilon_G(k)}\right), \quad (5.22)$$

where $\mathcal{E}_i$ was defined in (5.2) and $\sum_{i=1}^{k}\mathcal{E}_i \leq 1.1k^2/\Upsilon_k(G)$ by Lemma 4.1. Combining (5.20) with (5.21) and (5.22), we have that

$$\sum_{i=1}^{k}\sum_{u\in S_i} d_u \cdot \|x(u) - x(c_i)\|^2 = \widetilde{O}\left(\frac{k^3}{\Upsilon_G(k)}\right).$$

Moreover, by (5.10) and (5.11) it is straightforward to see that the distance between any embedded vertices decreases as we increase the value of t. Hence, the statement holds for any $t = \Omega\left(1/(\lambda_k \cdot \log n)\right)$. □

Lemma 5.9. *Let $A_1, \ldots, A_k$ be a k-way partition returned by Algorithm 3. Then, under a proper permutation of the indices, with constant probability for any $1 \leq i \leq k$ it holds that (i) $\mathrm{vol}(A_i \triangle S_i) = \widetilde{O}\left(k^3/\Upsilon_G(k)\right)\mathrm{vol}(S_i)$, and (ii) $\phi_G(A_i) \leq 1.1 \cdot \phi_G(S_i) + \widetilde{O}\left(k^3/\Upsilon_G(k)\right)$.*

Proof. We assume that $c_1, \ldots, c_k$ are the centres returned by SeedAndTrim when obtaining $A_1, \ldots, A_k$. By Lemma 5.8, with constant probability it holds that $\{c_1, \ldots, c_k\} \subseteq \bigcup_{i=1}^{k} \mathsf{CORE}_i^{\alpha}$, and c_i and c_j belong to different cores for $i \neq j$. Without loss of generality, we assume that $c_i \in \mathsf{CORE}_i^{\alpha}$. Then, it holds that

$$
\begin{aligned}
&\mathrm{vol}(S_i \setminus A_i) \\
&\leq \sum_{i \neq j} \mathrm{vol}\left(\left\{v \in S_i : \|x(c_i) - x(v)\| \geq \frac{\|x(c_j) - x(v)\|}{\log k}\right\}\right) \\
&\leq \sum_{i \neq j} \mathrm{vol}\left(\left\{v \in S_i : \|x(c_i) - x(v)\| \geq \frac{\|x(c_i) - x(c_j)\| - \|x(c_i) - x(v)\|}{\log k}\right\}\right) \\
&\leq \sum_{i \neq j} \mathrm{vol}\left(\left\{v \in S_i : 2\|x(c_i) - x(v)\| \geq \frac{\|x(c_i) - x(c_j)\|}{\log k}\right\}\right) \\
&\leq \sum_{i \neq j} \mathrm{vol}\left(\left\{v \in S_i : \|x(c_i) - x(v)\|^2 = \widetilde{\Omega}\left(\frac{1}{\min\{\mathrm{vol}(S_j), \mathrm{vol}(S_i)\}}\right)\right\}\right) \qquad (5.23) \\
&= \widetilde{O}\left(k^3/\Upsilon_G(k)\right)\mathrm{vol}(S_i), \qquad (5.24)
\end{aligned}
$$

where 5.23 follows from the second statement of Lemma 5.8.

Similarly, we also have that

$$
\begin{aligned}
\mathrm{vol}(A_i \setminus S_i) &\leq \sum_{i \neq j} \mathrm{vol}\left(\left\{v \in S_j : \|x(c_j) - x(v)\| \geq \frac{\|x(c_i) - x(v)\|}{\log k}\right\}\right) \\
&= \widetilde{O}\left(k^3/\Upsilon_G(k)\right)\mathrm{vol}(S_i).
\end{aligned}
$$

This yields the first statement of the lemma. The second statement follows by the same argument used in proving Theorem 4.1. □

The approximation guarantee of the returned partition is shown in Lemma 5.9. For the runtime, notice that the algorithm enumerates at most $O(\mathrm{poly}\log n)$ possible values

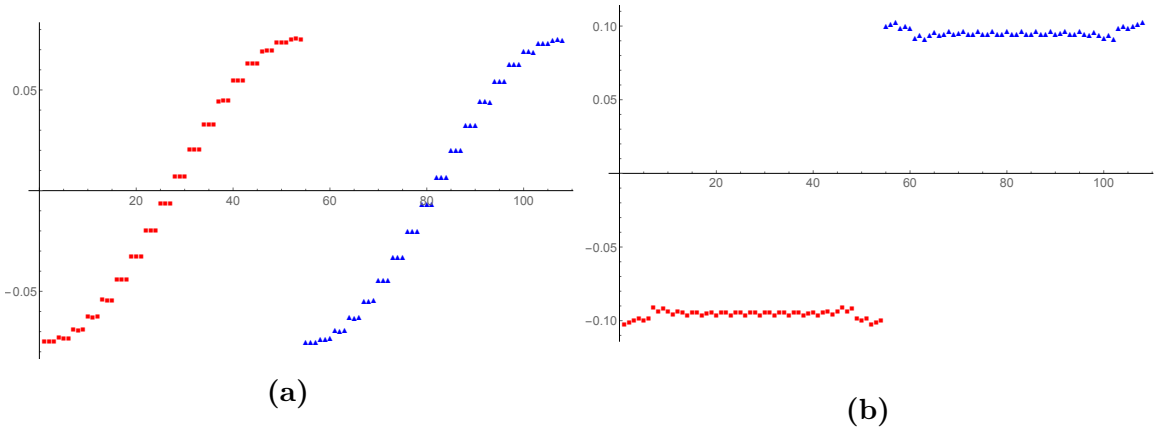
of t, and for every such value of t the algorithm runs in $\widetilde{O}(m)$ time, which includes computing the distances of the embedded points, and the seeding and grouping steps. Therefore, the total runtime is $\widetilde{O}(m)$. This proves Theorem 5.1.

5.3 Further discussions

In Lemma 5.7 we have shown that the heat-kernel embedding $\{x_t(u)\}_{u\in V}$, for appropriate choices of the temperature parameter t, approximates the spectral embedding of well-clustered graphs. This follows from the fact that a large enough value of $\Upsilon_G(k)$ implies a gap between λ_k and λ_{k+1}. Therefore, from (5.10) it is clear that an appropriate choice of t makes the contribution of the higher eigenvectors negligible. On the other hand, for graphs in which there is no significant gap between consecutive eigenvalues, the heat kernel distance takes the contribution from all eigenvectors into account. Does it overcome the barriers faced by classical spectral clustering algorithms and potentially provide us with a new approach to design graph clustering algorithms? To study this question, we will revisit the graph Grid of Figure 3.3.

Recall the graph Grid in Figure 3.3 has a clear optimal partitioning, i.e., the one cutting the grid horizontally, but has a small value of $\Upsilon_{\mathsf{Grid}}(2)$, and indeed the second eigenvector of $\mathcal{L}_{\mathsf{Grid}}$ produces a bad partitioning. If we compute the heat-kernel embedding of Grid for a suitable parameter t, however, we can see the following interesting phenomenon arising: suppose $\{u, v\}$ is an edge of Grid so that u belongs to one side of the optimal cut and v to the other, while $\{w, z\}$ connects two vertices on the same side of the optimal cut. Then, for a proper choice of the parameter t, we have that $\eta_t(u, v) \gg \eta_t(w, z)$, i.e., the heat-kernel distance between u and v will be much larger than the heat kernel distance between w and z. This essentially says that the heat-kernel is able to distinguish edges across the optimal cut. This also suggests the following partitioning algorithm, which is inspired by a reweighting scheme proposed in [70]. Let $\mathsf{Grid}^\star = (V, E, w^\star)$ be a graph defined on the same vertex and edge sets of Grid but with a different weight function $w^\star$: for any $\{u, v\} \in E$, $w^\star(u, v) = \exp(-c \cdot \eta_t(u, v))$ for a large enough constant $c > 1$. This weight function ensures that if the heat-kernel distance between two vertices u, v is large, then the weight of the edge between the two will be very small. Thanks to the properties of the heat-kernel, edges across the optimal cut will have weight very close to 0, and we directly apply the standard spectral partitioning algorithm on $\mathsf{Grid}^\star$: simulations have

shown this algorithm outputs the optimal cut for Grid*, which is also the optimal cut of Grid. See Figure 5.1 for an illustration of this phenomenon.



(a) **(b)**

Figure 5.1: The values of the second eigenvector of the Laplacian of Grid on 108 vertices, before and after reweighing. Here, the x-axes indicates the different vertices, while the y-axes indicates the corresponding values of the second eigenvector for each vertex. The colour of each point indicates which cluster the corresponding vertex belongs to. In (a), the plot of the second eigenvector of the Laplacian of Grid. In (b), the plot of the second eigenvector of the Laplacian of Grid after its edges have being reweighed according to the heat kernel distances between their endpoints, where $t = 6$ and $c = 100$. The figures clearly show that spectral partitioning on the original graph Grid does not produce the optimal clustering, while it does on the reweighed graph.

We have successfully tested this algorithm on several instances for which spectral methods notoriously fail. Unfortunately, we have not found a theoretical justification for its success. We conjecture the power of heat kernel in partitioning these well-known "bad instances" lies on the fact that these graphs are usually constructed in a way such that the indicator vector of the sparest cut becomes close to a linear combination of the eigenvectors corresponding to λ_k for $k \geq 3$, hence any spectral algorithm based on f_2, e.g., Algorithm 1, will certainly fail. The heat-kernel embedding, however, is much more stable than the spectral embedding: in fact, eigenvectors are in general highly susceptible to small perturbations of the graph, while it is much more difficult to perturb a graph so that the heat-kernel embedding changes, but the optimal cut doesn't [44]. It would be very interesting to prove there exists a natural class of graphs for which graph partitioning methods based on heat-kernel succeed even when spectral methods fail. We leave this as an open question for future work.

Chapter 6

Distributed graph clustering

In the previous chapters we studied clustering algorithms which have fast running time and are relatively simple to describe and implement. These algorithms, however, have one major drawback: they require data to be stored and processed in a single site. In this chapter we assume data is collected and processed at different sites, and communication between sites is allowed, but expensive. In particular, we consider the so-called `CONGEST` model of distributed computing [54], in which we have a network whose topology is described by an undirected graph G of n vertices and each node of the network (vertex in the graph) is a computational unit. The computation proceeds in synchronous rounds: in each round, nodes communicate with their neighbours with messages of size at most $O(\log n)$. Algorithms for the `CONGEST` model aim to solve a computational problem while minimising the number of rounds and the total communication required.

We consider the problem of partitioning a well-clustered graph in the `CONGEST` model. We assume the graph to be partitioned also describes the topology of the network, and we would like our algorithm to assign a label to each node so that nodes inside the same cluster will receive the same label, while nodes belonging to different clusters will receive different ones. Our algorithm comprises of two distinct procedures: the first one is a distributed sparsification procedure that takes as input a (dense) regular graph G and outputs an edge-induced subgraph $G^\star$ that is sparse and possesses the same cluster structure of G. This is crucial to reduce the communication between nodes required by the clustering algorithm. The second procedure consists in the actual clustering algorithm, which assigns each node a label that corresponds to the cluster the node belongs to. Under some reasonable assumptions, the algorithm works in

poly $\log n$ rounds and assigns the correct label to the majority of the nodes. Combining these two procedures, we obtain an algorithm which has the following guarantees.

Theorem 6.1. *There is a distributed algorithm that, given as input an n-vertex graph $G = (V, E)$ with k optimal clusters $S_1, \ldots, S_k$ such that $\mathrm{vol}(S_i) \geq \beta \,\mathrm{vol}(V)$ for any $1 \leq i \leq k$ and*

$$\Upsilon_G(k) = \omega\left(k^4 \log^2 \frac{1}{\beta} + \log n\right), \tag{6.1}$$

finishes in $T \triangleq \Theta\left(\log n/\lambda_{k+1}\right)$ rounds, and with constant probability the following statements hold:

1. *Each node v receives a label ℓ_v such that the total volume of misclassified nodes is $o(\mathrm{vol}(V))$, i.e., under a possible permutation of the labels σ, it holds that*

$$\mathrm{vol}\left(\bigcup_{i=1}^{k} \{v \colon v \in S_i \text{ and } \ell_v \neq \sigma(i)\}\right) = o(\mathrm{vol}(V));$$

2. *The total information exchanged among the n node, i.e., the message complexity, is $O\left(T^2 \cdot n \cdot \frac{1}{\beta} \log \frac{1}{\beta}\right)$ words.*

To emphasise the strength of Theorem 6.1, we look at the case where G consists of $k = O(1)$ regular expanders of balanced size connected by sparse cuts. Theorem 6.1 states we can recover the cluster-structure of G in $O(\log n)$ rounds so that the number of misclassified vertices is $o(n)$. Moreover, the total information exchanged among all nodes is $O(n \log^2 n)$ words, which is sublinear in the size of G for a dense input graph. Indeed, if we consider a sequential implementation of our algorithm, and we assume the algorithm have access to an oracle that, given a node, outputs one of its neighbours at random, our algorithm works in time that is nearly-linear in the number of vertices of G and, if G is dense, sublinear in the number of edges of G. Notice that, however, this distributed algorithm requires stronger assumptions on G than the sequential algorithms discussed in previous chapters. In particular it requires the optimal clusters to be balanced in size.

The chapter is organised as follows: in Section 6.1 we present our sparsification procedure, while we devote Section 6.2 to the presentation and analysis of the distributed clustering algorithm, which ends with the proof of Theorem 6.1. In Section 6.3 we give experimental results on the quality of the approximation output by our sparsification procedure. We end the chapter with some open problems.

6.1 The sparsification lemma

The first phase of our algorithm is a sparsification procedure whose aim is to reduce the connections among nodes while retaining the cluster-structure of the original network. It is inspired by known constructions of spectral sparsifiers, which are sparse approximations of graphs preserving their spectral properties [10]. In particular, our algorithm is similar to the construction of spectral sparsifiers for expander graphs [64], while its analysis is inspired by the construction of spectral sparsifiers via effective resistances [63]. Unlike these constructions, however, our algorithm does not output a graph which is spectrally equivalent to the one in input, but only preserves its cluster-structure. On the other hand, our algorithm is considerably simpler and inherently distributed, and the total communication between nodes required by our algorithm is only $O(n \operatorname{poly}\log n)$ on most networks. More precisely, we have the following result.

Lemma 6.1. *There exists a distributed algorithm that, receiving as input a graph $G = (V, E, w)$ with k clusters and a parameter τ such that $\tau \geq C/\lambda_{k+1}$ for a large enough constant $C > 0$, with constant probability, computes a sparsifier $H = (V, F \subset E)$ with $|F| = O(\tau\, n \log n)$ edges such that*

1. $\Upsilon_H = \Omega(\Upsilon_G(k)/k)$,
2. $\phi_H(S_i) = O(k \cdot \phi_G(S_i))$ *for any* $1 \leq i \leq k$

Moreover, this algorithm can be implemented in a single synchronous round, and the total information exchanged among all nodes is $O(\tau\, n \log n)$ words.

The first property of the output graph H in Lemma 6.1, $\Upsilon_H = \Omega(\Upsilon_G(k)/k)$, ensures that the gap in H is preserved as long as $\Upsilon_G(k) \gg k$. The second property further guarantees that the conductance of each optimal cluster S_i in G is approximately preserved in H up to a factor of k, therefore S_i is a low-conductance subset in H as well.

We remark that this optimal clustering $S_1, \ldots, S_k$ may not be optimal in H anymore. However, this is not an issue, since every cluster with low conductance in H has a large overlap with its optimal correspondence. This implies that any algorithm that recovers a clustering close to the optimal one in H will recover a clustering close to the optimal one in G as well. Moreover, since λ_{k+1} represents the inner-connectivity of the clusters, it is usually quite high: for most interesting cases we can reasonably assume $\lambda_{k+1} = \Omega\left(1/\operatorname{poly}(\log n)\right)$ and, therefore, both the communication cost and the

size of the output are quite small. Indeed, the experiments described in Section 6.3 show $\tau \leq 2$ works for all the tested datasets.

6.1.1 Algorithm

Our algorithm is based on sampling edges with respect to the degrees of their endpoints, which was originally introduced in [63] as a way to construct spectral sparsifiers for graphs with *high* spectral expansion. To sketch the intuition behind our algorithm, let us look at the following toy example illustrated in Figure 6.1, i.e., the graph G consisting of two complete graphs of n vertices connected by a single edge. It is easy to see that, when we sample $O(n \log n)$ edges uniformly at random from G to form a graph H, with high probability the middle edge will not be sampled and graph H will consist of two isolated graphs, each of which has constant spectral expansion. Although our sampled graph H does not preserve the spectral and cut structure of G, it does preserve its cluster-structure: every reasonable clustering algorithm will recover these two disjoint components of H, which correspond exactly to the two clusters in G. We will show that this sampling scheme can be generalised, and sampling every edge $\{u, v\}$ with probability depending only on d_u and d_v suffices to build a sparse subgraph that preserves the cluster-structure of the original graph.

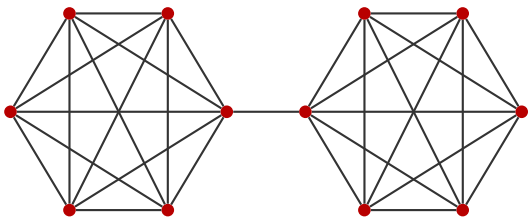


Figure 6.1: The graph G consists of two complete subgraphs of n vertices connected by an edge. It is easy to see that sampling $O(n \log n)$ edges uniformly at random suffices to obtain a subgraph having the same cluster-structure of G.

Formal Description: In our algorithm every vertex u checks every edge $e = \{u, v\}$ adjacent to u itself, and samples edge e with probability

$$p_u(v) \triangleq \min\left\{w(e) \cdot \frac{\tau \ \log n}{d_u}, 1\right\} \tag{6.2}$$

for some parameter τ satisfying $\tau \geq C/\lambda_{k+1}$ for a large enough constant $C > 0$. The algorithm uses a set F to maintain all the sampled edges, where we initially set $F = \emptyset$.

Finally, the algorithm returns a weighted graph $H = (V, F, w_H)$, where the weight $w_H(e)$ of every edge $e = \{u, v\} \in F$ is defined as $w_H(e) \triangleq \frac{w(e)}{p_e}$, and

$$p_e \triangleq p_u(v) + p_v(u) - p_u(v) \cdot p_v(u)$$

is the probability that e is sampled by at least one of its endpoints. Choosing the right parameter τ is application-dependent and will be discussed in Section 6.3. The pseudocode for the algorithm is listed in Algorithm 4.

Algorithm 4 Cluster-preserving sparsification

Input: graph $G = (V, E, w)$ and a parameter τ such that $\tau \geq C/\lambda_{k+1}$ for a large enough $C > 0$.
$F = \emptyset$
for every vertex $u \in V$ **do**
  for each edge $e = \{u, v\}$ adjacent to u **do**
    add e to F with probability $p_u(v)$ defined in (6.2)
for each edge $e = \{u, v\} \in F$ **do**
  $w_H(e) = w(e)/p_e$
Output: $H = (V, F, w_H)$

Distributed Nature of the Algorithm: Our algorithm can be easily implemented in a distributed way: any vertex u chooses to retain (or not) an edge $e = \{u, v\}$ independently from any other vertex, and communication between u and v is needed only if e is sampled by one of its two endpoints. Therefore, the total communication needed is proportional to the number of edges in H.

6.1.2 Analysis of the algorithm

Now we analyse the algorithm, and prove Lemma 6.1. At a high level, our proof consists of the following two steps:

1. We analyse the intra-connectivity of the clusters in the returned graph H: we show that the top $n - k$ eigenspaces of $\mathcal{L}_G$ are preserved in $\mathcal{L}_H$, and hence $\lambda_{k+1}(\mathcal{L}_H) = \Theta(\lambda_{k+1}(\mathcal{L}_G))$.

2. We show that the conductance of $S_1, \ldots, S_k$ are low in H, i.e.,

$$\phi_H(S_i) = O\left(k \cdot \phi_G(S_i)\right) \text{ for any } i = 1, \ldots, k. \tag{6.3}$$

Combining these two steps, we will prove that $\Upsilon_H = \Omega(\Upsilon_G(k)/k)$, which proves the approximation guarantees of Lemma 6.1. The bound on the number of edges in H follows from the definition of the sampling scheme of our algorithm.

The following concentration inequalities will be used in our proof.

Lemma 6.2 (Problem 1.9, [23])**.** *Given n independent random variables $X_1, \ldots, X_n$ such that $x_i \in [a_i, b_i]$ for any $i \in [n]$, let $X = \sum_{i=1}^n X_i$. Then it holds that*

$$\mathbb{P}[|X - \mathbf{E}[X]| \geq t] \leq 2\exp\left(-\frac{2t^2}{\sum_i (b_i - a_i)^2}\right).$$

Lemma 6.3 (Matrix Chernoff Bound, [72])**.** *Consider a finite sequence $\{X_i\}$ of independent, random, PSD matrices of dimension d that satisfy $\|X_i\| \leq R$. Let $\mu_{\min} \triangleq \lambda_{\min}(\mathbb{E}[\sum_i X_i])$ and $\mu_{\max} \triangleq \lambda_{\max}(\mathbb{E}[\sum_i X_i])$. Then it holds that*

$$\mathbb{P}[\lambda_{\min}\left(\sum_i X_i\right) \leq (1-\delta)\mu_{\min}] \leq d \cdot \left(\frac{\mathrm{e}^{-\delta}}{(1-\delta)^{1-\delta}}\right)^{\mu_{\min}/R} \quad \text{for } \delta \in [0,1], \text{ and}$$

$$\mathbb{P}[\lambda_{\max}\left(\sum_i X_i\right) \geq (1+\delta)\mu_{\max}] \leq d \cdot \left(\frac{\mathrm{e}^{\delta}}{(1+\delta)^{1+\delta}}\right)^{\mu_{\max}/R} \quad \text{for } \delta \geq 0.$$

Proof of Lemma 6.1. Without loss of generality we assume within the proof that it holds for any edge $e = \{u, v\}$ that

$$w(u,v) \cdot \frac{\tau \cdot \log n}{d_u} \leq 1. \tag{6.4}$$

Notice that otherwise e is always added into H by Algorithm 4. For the sake of our analysis, we can then "split" the edge e into $r \geq 1$ copies $e_1, \ldots, e_r$ (we can always assume r to be integer, since we can adjust τ by a constant factor without affecting the analysis) such that $w(e_i) = w(e)/r$ and $w(e_1) \cdot \frac{\tau \cdot \log n}{d_u} = 1$: Algorithm 4 will still add all the copies $e_1, \ldots, e_r$ to H with probability 1, but now (6.4) will be satisfied for each edge. In the rest of the proof we will then assume (6.4) is satisfied.

Let $\overline{\mathcal{L}}_G$ be the projection of $\mathcal{L}_G$ on its top $n-k$ eigenspaces, i.e., $\overline{\mathcal{L}}_G = \sum_{i=k+1}^n \lambda_i f_i f_i^\intercal$. With a slight abuse of notation we denote by $\overline{\mathcal{L}}_G^{-1/2}$ the square root of the pseudoinverse of $\overline{\mathcal{L}}_G$, i.e.,

$$\overline{\mathcal{L}}_G^{-1/2} = \sum_{i=k+1}^{n} (\lambda_i)^{-1/2} f_i f_i^\intercal.$$

Analogously, we call $\overline{\mathcal{I}}$ the projection on $\mathrm{span}\{f_{k+1}, \ldots, f_n\}$, i.e.., $\overline{\mathcal{I}} \triangleq \sum_{i=k+1}^n f_i f_i^\intercal$.

We will first prove that the top $n-k$ eigenspaces of $\mathcal{L}_G$ are preserved, which implies that $\lambda_{k+1}(\mathcal{L}_G) = \Theta(\lambda_{k+1}(\mathcal{L}_H))$. To prove this statement, we examine the properties of the graph H constructed by the algorithm. Remember that for any $e = \{u, v\}$ we have $p_e = p_u(v) + p_v(u) - p_u(v) \cdot p_v(u)$, and it holds that $\frac{1}{2}(p_u(v) + p_v(u)) \leq p_e \leq p_u(v) + p_v(u)$. Also recall that $\mathcal{L}_G = \sum_{e \in E} w(e) c_e c_e^\intercal$, where for an edge $e = \{u, v\}$, c_e is defined as

$$c_e(z) = \begin{cases} 1/\sqrt{d_u} & \text{if } z = u \\ -1/\sqrt{d_v} & \text{if } z = v \\ 0 & \text{otherwise} \end{cases}$$

where the role of u and v can be interchanged. We then define a random matrix $X_e \in \mathbb{R}^{n \times n}$ by

$$X_e = \begin{cases} w_H(u, v) \cdot \overline{\mathcal{L}}_G^{-1/2} c_e c_e^\intercal \overline{\mathcal{L}}_G^{-1/2} & \text{if } u \sim v \text{ is sampled by the algorithm,} \\ 0 & \text{otherwise.} \end{cases}$$

Notice that

$$\begin{aligned} \sum_{e \in E[G]} X_e &= \sum_{\text{sampled edges } e = \{u,v\}} w_H(u, v) \cdot \overline{\mathcal{L}}_G^{-1/2} c_e c_e^\intercal \overline{\mathcal{L}}_G^{-1/2} = \overline{\mathcal{L}}_G^{-1/2} \mathcal{L}_H \overline{\mathcal{L}}_G^{-1/2}, \\ \mathbb{E}[\sum_{e \in E} X_e] &= \sum_{e = \{u,v\} \in E[G]} p(e) \cdot w_H(u, v) \cdot \overline{\mathcal{L}}_G^{-1/2} c_e c_e^\intercal \overline{\mathcal{L}}_G^{-1/2} \\ &= \overline{\mathcal{L}}_G^{-1/2} \mathcal{L}_G \overline{\mathcal{L}}_G^{-1/2} = \overline{\mathcal{I}}. \end{aligned}$$

Moreover, for any sampled $e = \{u, v\} \in E$ we have that

$$\begin{aligned} \|X_e\| &= w_H(u, v) \cdot \left\| \overline{\mathcal{L}}_G^{-1/2} c_e c_e^\intercal \overline{\mathcal{L}}_G^{-1/2} \right\| \\ &= w_H(u, v) \cdot c_e^\intercal \overline{\mathcal{L}}_G^{-1/2} \overline{\mathcal{L}}_G^{-1/2} c_e \\ &= \frac{w(u, v)}{p_e} \cdot c_e^\intercal \overline{\mathcal{L}}_G^{-1} c_e \\ &\leq \frac{w(u, v)}{p_e} \left\| \overline{\mathcal{L}}_G^{-1} \right\| \|c_e\|^2 \\ &\leq \frac{2}{\tau \log n \cdot \left(\frac{1}{d_u} + \frac{1}{d_v} \right)} \cdot \frac{1}{\lambda_{k+1}} \left(\frac{1}{d_u} + \frac{1}{d_v} \right) \\ &\leq \frac{2}{C \log n}, \end{aligned}$$

where the second equality follows from the trivial fact that, for any vector x, $\|xx^\intercal\| = x^\intercal x$, while the last inequality follows by $\tau \geq C/\lambda_{k+1}$. To apply the matrix Chernoff bound, we need to restrict ourselves to the top $(n-k)$ eigenspaces, i.e., we assume that $\overline{\mathcal{I}}$ is the identity of this $(n-k)$-dimensional space, and therefore has all its eigenvalues equal to one. We can then apply Lemma 6.3 with $\mu_{\min} = \mu_{\max} = 1$, $R = \frac{2}{C \log n}$, and $\delta = 1/2$. We obtain that

$$\mathbf{P}\left[\lambda_{\min}\left(\sum_{e \in E[G]} X_e\right) \geq 1/2\right] = 1 - O(1/n^c),$$

and

$$\mathbf{P}\left[\lambda_{\max}\left(\sum_{e \in E[G]} X_e\right) \leq 3/2\right] = 1 - O(1/n^c),$$

for some large constant c. Combining this with the Courant-Fischer theorem (2.2) and $\dim(\text{span}\{f_{k+1}, \ldots, f_n\}) = n - k$, we have that $\lambda_{k+1}(\mathcal{L}) = \Theta(\lambda_{k+1}(\mathcal{L}_G))$.

Now we analyse the conductance of every cluster S_i in H. For any edge $e = \{u, v\}$ we define a random variable such that

$$Y_e = \begin{cases} \frac{w(u,v)}{p_e} & \text{with probability } p_e, \\ 0 & \text{otherwise.} \end{cases}$$

Hence, it holds for any $1 \leq i \leq k$ that

$$\mathbb{E}[w_H(S_i, V \setminus S_i)] = \mathbb{E}\left[\sum_{\substack{e=\{u,v\}, \\ u \in S_i, v \notin S_i}} Y_{\{u,v\}}\right] = w(S_i, V \setminus S_i).$$

Hence, by Markov's inequality and the union bound, with constant probability it holds for all $i = 1, \ldots, k$ that

$$w_H(S_i, V \setminus S_i) = O\left(k \cdot \delta_G(S_i)\right). \tag{6.5}$$

We further analyse $\text{vol}_H(S_i)$. For any $u \in S_i$, let

$$d^\star_G(u) = \sum_{\substack{u \sim v, v \in S_i \\ d_u \leq d_v}} w(u, v),$$

and

$$\mathrm{vol}^\star_G(S_i) = \sum_{u \in S_i} d^\star_G(u).$$

Since $w(u,v)$ for any internal edge $u \sim v$ in S_i contributes only twice to $\mathrm{vol}_G(S_i)$ and $\phi_G(S_i) \leq 1/2$, it always holds that $\mathrm{vol}^\star_G(S_i) \geq \mathrm{vol}_G(S_i)/4$. We define random variables $d^\star_H(u) = \sum_{u \sim v, v \in S_i} Y_e$, and $\mathrm{vol}^\star_H(S_i) = \sum_{u \in S_i} d^\star_H(u)$. By definition, we have that $\mathbb{E}[\mathrm{vol}^\star_H(S_i)] = \mathrm{vol}^\star_G(S_i)$. Since it holds for any $e = \{u,v\}$ that

$$0 \leq \frac{w(u,v)}{p_e} \leq \frac{2w(u,v)}{p_u + p_v} = \frac{2}{\tau \log n \cdot (1/d_u + 1/d_v)},$$

we can apply Lemma 6.2 and obtain that

$$\begin{aligned}
&\mathbb{P}[|\mathrm{vol}^\star_H(S_i) - \mathrm{vol}^\star_G(S_i)| \geq 1/2 \cdot \mathrm{vol}^\star_G(S_i)] \\
&\quad = 2 \cdot \exp\left(- \frac{1/2 \cdot (\mathrm{vol}^\star_G(S_i))^2}{\sum_{u,v \in S_i \colon u \sim v, d_u \leq d_v} \left(\frac{2}{\tau \log n \cdot (1/d_u + 1/d_v)} \right)^2} \right) \\
&\quad \leq 2 \cdot \exp\left(- \frac{\tau^2 \log^2 n \cdot (\sum_{u \in S_i} d_u)^2/100}{\sum_{u,v \in S_i \colon u \sim v, d_u \leq d_v} \left(\frac{d_u d_v}{d_u + d_v} \right)^2} \right) \\
&\quad \leq 2 \cdot \exp\left(- \frac{\tau^2 \log^2 n \cdot (\sum_{u \in S_i} d_u)^2/100}{\sum_{u,v \in S_i \colon u \sim v, d_u \leq d_v} d_u d_v} \right) \\
&\quad = O(1/n^c)
\end{aligned}$$

for some constant $c \geq 10$. Hence, with probability $1 - O(1/n^c)$ it holds that

$$\mathrm{vol}^\star_H(S_i) \geq \frac{1}{2} \cdot \mathrm{vol}^\star_G(S_i) \geq \frac{1}{8} \cdot \mathrm{vol}_G(S_i). \tag{6.6}$$

By the union bound, (6.6) holds for all the k clusters. Combining this with (6.5) shows that $\phi_H(S_i) = O(k \cdot \phi_G(S_i))$ for any $1 \leq i \leq k$, and $\Upsilon_H = \Omega(\Upsilon_G(k)/k)$. □

6.2 Clustering algorithm

In this section we describe and analyse the clustering algorithm. The aim is to prove the following theorem which, combined with Lemma 6.1, implies Theorem 6.1.

Theorem 6.2. *There is a distributed algorithm that, given as input a graph $G = (V, E, w)$ with n vertices, m edges, and k optimal clusters $S_1, \ldots, S_k$ with $\mathrm{vol}(S_i) \geq$*

$\beta \operatorname{vol}(V)$ *for any* $1 \leq i \leq k$ *and*

$$\Upsilon_G(k) = \omega\left(k^4 \log^2 \frac{1}{\beta} + \log n\right), \tag{6.7}$$

finishes in $T \triangleq \Theta\left(\log n/\lambda_{k+1}\right)$ *rounds, and with constant probability the following statements hold:*

1. *Each vertex* v *receives a label* ℓ_v *such that the total volume of misclassified vertices is* $o(\operatorname{vol}(V))$*, i.e., under a possible permutation of the labels* σ*, it holds that* $\operatorname{vol}\left(\bigcup_{i=1}^{k} \{v | v \in S_i \text{ and } \ell_v \neq \sigma(i)\}\right) = o(\operatorname{vol}(V))$*;*
2. *The total information exchanged among these* n *vertices, i.e., the message complexity, is* $O\left(T \cdot m \cdot \frac{1}{\beta} \log \frac{1}{\beta}\right)$ *words.*

We remark that the algorithm presented in this section is not the first distributed algorithm for graph clustering. Becchetti et al. [11], for example, study a distributed dynamic to partition an almost-regular graph into clusters, but their analysis focuses mostly on graphs generated randomly from stochastic block models. The design and analysis of our algorithm succeeds to overcome their regularity constraint thanks to an alternative averaging rule.

We also notice that the distributed algorithm presented in Kempe et al. [35] for computing the top k eigenvectors of the adjacency matrix of a graph can be applied for graph clustering. Their algorithm, however, is much more involved than ours. Moreover, for an input graph G of n nodes, the number of rounds required in their algorithm is proportional to the mixing time of a random walk in G. For a graph consisting of multiple expanders connected by very few edges, their algorithm requires $O(\operatorname{poly}(n))$ rounds, which is much higher than $O(\operatorname{poly}\log n)$ rounds needed for our algorithm.

In the next subsection we give a formal description of the algorithm. Later we present a more abstract view of the same.

6.2.1 Description

At the initialisation step, every node v picks a random number from 1 to n^3, which is used as the identification of node v. It is easy to show that, with high probability, all the nodes pick different numbers. We assume that this holds in the remaining part of

this chapter, and use $\mathsf{ID}(v)$ to represent the ID of node v. Our algorithm consists of three procedures:

The Seeding Procedure: Let $\bar{s} \triangleq \frac{3}{\beta}\ln\frac{1}{\beta}$. Each node v becomes *active* with probability $\bar{s}\cdot d_v/\operatorname{vol}(V)$. For each active node v, node v sets its initial state as $\mathsf{State}_v(0) = \{(\mathsf{ID}(v), 1/\sqrt{d_v})\}$. Every non-active node v sets $\mathsf{State}_v(0) = \emptyset$. We call, respectively, $\mathsf{ID}(v)$ and x the *prefix* and *suffix* of $(\mathsf{ID}(v), x)$. For any $v \in V$ and $t \geq 0$, $\mathsf{State}_v(t)$ is a set of pairs prefix/suffix that represents the *state* of node v at round t. A state cannot contain multiple pairs wth the same prefix. For simplicity, we set $\mathsf{State}_v(t)[\mathsf{ID}(x)] = y$ if $(\mathsf{ID}(x), y) \in \mathsf{State}_v(t)$, and we assume $\mathsf{State}_v(t)[\mathsf{ID}(x)] = 0$ if $(\mathsf{ID}(x), y) \notin \mathsf{State}_v(t)$.

The Averaging Procedure: The averaging procedure proceeds for T rounds. In each round $t \geq 1$, each node v computes its new state state $\mathsf{State}_v(t)$ based on its current state and the state of its neighbours. More formally, let $\mathsf{ID}(w_1), \ldots, \mathsf{ID}(w_\ell)$ be the list of prefixes appearing in the state of v or the state of at least one of its neighbours after round $t-1$. Then, v updates its state according to the following local rule. For any $i = 1, \ldots, \ell$,

$$\mathsf{State}_v(t)[\mathsf{ID}(w_i)] = \frac{1}{2}\mathsf{State}_v(t-1)[\mathsf{ID}(w_i)] + \frac{1}{2}\sum_{\{u,v\}\in E}\frac{1}{\sqrt{d_u d_v}}\mathsf{State}_u(t-1)[\mathsf{ID}(w_i)].$$

The Query Procedure: The query procedure assigns every node v a label ℓ_v, and we say that two nodes u, v are assigned by the algorithm to the same cluster if and only if $\ell_u = \ell_v$. Formally, based on $\mathsf{State}_v(T)$ node v chooses

$$\ell_v = \min\left\{\mathsf{ID}(w) \mid (\mathsf{ID}(w), x) \in \mathsf{State}_v(T) \bigwedge x \geq \frac{\sqrt{d_v}}{2\beta\operatorname{vol}(V)}\right\}$$

as the label of the cluster it belongs to, and ℓ_v is set to be an arbitrary ID if there is no vector $(\mathsf{ID}(w), x) \in \mathsf{State}_v(T)$ satisfying $x \geq \sqrt{d_v}/\left(2\beta\operatorname{vol}(V)\right)$.

6.2.2 Abstract formulation

From the description above, it is clear that the prefix of any vector is only used to identify from which active node that vector originated, and vectors with different prefixes will not be balanced together during the execution of the algorithm.

In the seeding procedure each node v becomes *active* with probability $\bar{s}\cdot d_v/\operatorname{vol}(V)$, where $\bar{s} \triangleq (3/\beta)\ln(1/\beta)$. For simplicity, we use s to denote the number of active nodes

at the end of these $\bar{s}$ trials, and $v_1, \cdots, v_s$ to denote these active nodes. Moreover, we introduce s vectors $x^{(0,1)}, \ldots, x^{(0,s)} \in \mathbb{R}^n$, where $x^{(0,i)}(v_i) = 1/\sqrt{d_{v_i}}$ and $x^{(0,i)}(u) = 0$ for any $u \neq v_i$. With a slight abuse of notation, we use χ_v to denote a vector such that $\chi_v(v) = 1/\sqrt{d_v}$ and $\chi_v(u) = 0$ for any $u \neq v$. Therefore, we can write $x^{(0,i)} = \chi_{v_i}$ for any $i = 1, \ldots, s$.

After that, the averaging procedure continues for T rounds, where in each round t the state of the nodes is updated according to the following rule:

$$x^{(t,i)} = \left(\frac{1}{2}\mathbb{I} + \frac{1}{2}D_G^{-1/2}A_G D_G^{-1/2}\right) x^{(t-1,i)} \qquad \text{for any } i = 1, \ldots, s \tag{6.8}$$

which can also be written as

$$x^{(t,i)} = \left(\mathbb{I} - \frac{1}{2}\mathcal{L}_G\right) x^{(t-1,i)} \qquad \text{for any } i = 1, \ldots, s \tag{6.9}$$

At the end of the algorithm, in the query procedure each node v checks its coordinates $x^{(T,1)}(v), \ldots, x^{(T,s)}(v)$, and uses

$$\ell_v = \min\left\{ i \mid x^{(T,i)}(v) \geq \frac{\sqrt{d_v}}{2\beta \operatorname{vol}(V)} \right\} \tag{6.10}$$

as the label of the cluster it chooses to belong. If no such index i exists, node v chooses an arbitrary label $\ell_v \in \{1, \ldots, n^3\}$.

The pseudocode of the algorithm is included in Algorithm 5.

Algorithm 5 A distributed algorithm for graph clustering

Let $\bar{s} = \Theta\left(\frac{1}{\beta}\log\frac{1}{\beta}\right)$
for $v \in V$ **do**
  v is an initial node with probability $\bar{s} \cdot d_v/\operatorname{vol}(V)$
Let $v_1, \ldots, v_s$ be the initial nodes.
for $i = 1, \ldots, s$ **do**
  $x^{(0,i)} = \chi_{v_i}$
for $t = 1, \ldots, T$ **do**
  for $i = 1, \ldots, s$ **do**
    for $v \in V$ **do**
      $x^{(t,i)}(v) = \frac{1}{2}x^{(t-1,i)}(v) + \frac{1}{2}\sum_{\{u,v\}\in E} \frac{w(u,v)}{\sqrt{d_u d_v}} x^{(t-1,i)}(u)$
for $v \in V$ **do**
  $\ell_v = \min\left\{ i \mid x^{(t,i)}(v) \geq \frac{\sqrt{d_v}}{2\beta \operatorname{vol}(V)} \right\}$
  Assign v to the cluster ℓ_v

6.2.3 Analysis

Before analysing the algorithm, we first discuss some intuitions behind the proof. Remember that the configuration of the network in round t of the averaging step is expressed by s vectors $x^{(t,1)}, \ldots, x^{(t,s)}$, and these vectors are updated according to (6.8). For the sake of intuition, we assume that G is regular. Then, the vector $x^{(t,i)}$ corresponds to the probability distribution of a t-step lazy random walk in G. It is well-known that the vector $x^{(t,i)}$ converges to the uniform distribution as t tends to infinity. The time $T = \Theta(\log n/\lambda_{k+1})$, instead, corresponds to the *local mixing time* of the clusters: if a random walk starts with $v_i \in S_i$, then the probability distribution of this T-step random walk will be mixed (uniform) inside S_i, conditioned on the fact that the random walk never leaves that cluster. Our analysis shows that, when picking v_i at random from S_i, with high probability the distribution of the random walk after T steps is concentrated on S_i. In other words, after T rounds, each vector $x^{(T,1)}, \ldots, x^{(T,s)}$ is almost uniform on one of the clusters, and close to zero everywhere else. Hence, as long as we hit all the clusters with at least one initial active node, the query step will assign the same label to two nodes if and only if they belong to the same cluster (for most pairs of nodes).

When G is not regular, (6.9) suggests that the averaging step can be thought as a *power iteration method* to approximate (k linearly independent combination of) the bottom eigenvectors of $\mathcal{L}_G$. We will show that these eigenvectors contain all the information needed to obtain a good partitioning of the graph.

We first analyse the averaging procedure on a single vector $x^{(0)}$. We show that after T rounds $x^{(T)}$ is close to the projection of $x^{(0)}$ on the bottom k eigenspaces of $\mathcal{L}_G$. This follows from the fact that there exists a gap between the bottom k eigenvalues of $\mathcal{L}_G$ and the rest of the spectrum. Applying (6.9) T times the contribution of the higher eigenspaces of $\mathcal{L}_G$ to $x^{(0)}$ becomes negligible, while the contribution of the bottom ones is not significantly affected. Similar with the definition of $\overline{\mathcal{I}}$, let $\underline{\mathcal{I}} = \sum_{i=1}^{k} f_i f_i^\intercal$ be the projection on the bottom k eigenspaces. We first prove that, starting the process with a single initial vector $x^{(0)}$, $x^{(T)}$ is close to $\underline{\mathcal{I}} x^{(0)}$.

Lemma 6.4. *For a large constant $c > 0$, it holds*

$$\left\| x^{(T)} - \underline{\mathcal{I}} x^{(0)} \right\| = O\left(\frac{\log n}{\Upsilon_G(k)} \cdot \left\| \underline{\mathcal{I}} x^{(0)} \right\| + n^{-c} \right).$$

Proof. By the update rule of the algorithm, we have that $x^{(T)} = P^T x^{(0)}$, where

$$P = \frac{1}{2} \cdot \mathbb{I} + \frac{1}{2} \cdot D^{-1/2} A D^{-1/2} = \mathbb{I} - \frac{1}{2}\mathcal{L}_G.$$

Hence, it holds that

$$P^T = \left(\mathbb{I} - \frac{1}{2}\mathcal{L}_G\right)^T = \sum_{i=1}^{n} \left(1 - \frac{\lambda_i}{2}\right)^T f_i f_i^\intercal.$$

Notice that for $i > k$ it holds that

$$\left(1 - \frac{\lambda_i}{2}\right)^T \leq \left(1 - \frac{\lambda_{k+1}}{2}\right)^{2c \cdot \log n / \lambda_{k+1}} \leq \mathrm{e}^{-c \log n} \leq n^{-c},$$

while for $i = 1, \ldots, k$ it holds that

$$\left(1 - \frac{\lambda_i}{2}\right)^T \geq 1 - \frac{T \cdot \lambda_i}{2} = 1 - O\left(\frac{\log n \cdot \lambda_i}{\lambda_{k+1}}\right) = 1 - O\left(\frac{\log n \cdot \lambda_k}{\lambda_{k+1}}\right).$$

Hence,

$$\begin{aligned}
\left\|x^{(T)} - \underline{\mathcal{I}} x^{(0)}\right\|^2 &= \left\|\left(P^T - \underline{\mathcal{I}}\right) x^{(0)}\right\|^2 \\
&= \sum_{i=1}^{k} \left(1 - \left(1 - \frac{\lambda_i}{2}\right)^T\right)^2 \left\langle f_i, x^{(0)} \right\rangle^2 + \sum_{i=k+1}^{n} \left(1 - \frac{\lambda_i}{2}\right)^{2T} \left\langle f_i, x^{(0)} \right\rangle^2 \\
&= O\left(\left(\frac{\log n \cdot \lambda_k}{\lambda_{k+1}}\right)^2 \sum_{i=1}^{k} \left\langle f_i, x^{(0)} \right\rangle^2 + n^{-2c}\right) \\
&= O\left(\left(\frac{\log n \cdot \lambda_k}{\lambda_{k+1}} \left\|\underline{\mathcal{I}} x^{(0)}\right\|\right)^2 + n^{-2c}\right) \\
&= O\left(\left(\frac{\log n}{\Upsilon_G k} \left\|\underline{\mathcal{I}} x^{(0)}\right\|\right)^2 + n^{-2c}\right),
\end{aligned}$$

where the last inequality follows from the higher-order Cheeger inequality [39]. Then, taking the square root on both sides of the equality above proves the lemma. □

As the goal is to use $x^{(T)}$ to recover the clusters, we need to relate $\underline{\mathcal{I}}$ to their indicator vectors. The following result is a corollary of Theorem 3.1.

Lemma 6.5. *Let $\Upsilon_G(k) = \Omega(k^2)$. For any $1 \leq i \leq k$ there exists $\widetilde{\chi}_i \in \operatorname{span}\{\overline{\chi}_{S_1}, \ldots, \overline{\chi}_{S_k}\}$ such that $\|\widetilde{\chi}_i - f_i\| = O\left(k\sqrt{\frac{k}{\Upsilon_G(k)}}\right)$. Moreover $\{\widetilde{\chi}_i\}_{i=1}^k$ form an orthonormal set.*

Proof of Lemma 6.5. Theorem 3.1 gives us a set of vectors $\{\widehat{\chi}_i\}_{i=1}^k$ which are linear combinations of the indicator vectors of the clusters and that well-approximate the bottom k eigenvectors of $\mathcal{L}_G$. Notice that since $\{f_i\}_{i=1}^k$ is an orthonormal set, $\{\widehat{\chi}_i\}_{i=1}^k$ are *almost* orthonormal as well. Therefore, our task is to construct an orthonormal set $\{\widetilde{\chi}_i\}_{i=1}^k$ based on $\{\widehat{\chi}_i\}_{i=1}^k$, which can be achieved by applying the Gram-Schmidt orthonormalisation procedure. The error bound follows from the fact that

$$\langle \widehat{\chi}_i, \widehat{\chi}_j \rangle = \frac{1}{2} \cdot \left(\|\widehat{\chi}_i\|^2 + \|\widehat{\chi}_j\|^2 - \|\widehat{\chi}_i - \widehat{\chi}_j\|^2 \right) = O\left(\sqrt{\frac{k}{\Upsilon_G(k)}} \right)$$

holds for $i \neq j$. □

Based on Lemma 6.5, we will prove in the next lemma that, for any cluster $S_1, \ldots, S_k$ and for most starting vertices $v \in S_j$, $x^{(T)}$ is close to $\overline{\chi}_{S_j}$.

Lemma 6.6. *Let $A \subseteq V$ be the subset of vertices such that, for any $j = 1, \ldots, k$ and any $v \in A \cap S_j$, setting $x^{(0)} = \chi_v$ we have that*

$$\left\| x^{(T)} - \frac{1}{\sqrt{\mathrm{vol}(S_j)}} \overline{\chi}_{S_j} \right\| = O\left(\sqrt{\frac{k^4 \log(1/\beta)}{\Upsilon_G(k) \beta \,\mathrm{vol}(V)}} \right).$$

Then, it holds that

$$\mathrm{vol}(A) \geq \mathrm{vol}(V) \left(1 - \frac{\beta}{C \log(1/\beta)} \right),$$

for some constant C.

Proof. Without loss of generality we assume $v \in S_j$, and let $\{\widetilde{\chi}_i\}_{i=1}^k$ be the set of vectors defined in Lemma 6.5. We show that the projection of χ_v on $\mathrm{span}\{\widetilde{\chi}_1, \ldots, \widetilde{\chi}_k\}$ is exactly equal to $\frac{1}{\sqrt{\mathrm{vol}(S_j)}} \overline{\chi}_{S_j}$. To this end, first notice that $\mathrm{span}\{\widetilde{\chi}_1, \ldots, \widetilde{\chi}_k\} = \mathrm{span}\{\overline{\chi}_{S_1}, \ldots, \overline{\chi}_{S_k}\}$, since each $\widetilde{\chi}_i$ is by definition a linear combination of vectors in $\{\overline{\chi}_{S_i}\}_{i=1}^k$ and

$$\dim\left(\mathrm{span}\{\widetilde{\chi}_1, \ldots, \widetilde{\chi}_k\}\right) = \dim\left(\mathrm{span}\{\overline{\chi}_{S_1}, \ldots, \overline{\chi}_{S_k}\}\right) = k.$$

Then,

$$\sum_{i=1}^k \langle \chi_v, \widetilde{\chi}_i \rangle \, \widetilde{\chi}_i = \sum_{i=1}^k \left\langle \chi_v, \overline{\chi}_{S_i} \right\rangle \overline{\chi}_{S_i} = \left\langle \chi_v, \overline{\chi}_{S_j} \right\rangle \overline{\chi}_{S_j} = \frac{1}{\sqrt{\mathrm{vol}(S_j)}} \overline{\chi}_{S_j}. \tag{6.11}$$

where the first equality holds by the fact that

$$\operatorname{span}\{\widetilde{\chi}_1, \ldots, \widetilde{\chi}_k\} = \operatorname{span}\{\overline{\chi}_{S_1}, \ldots, \overline{\chi}_{S_k}\}$$

and the orthonormality of the two sets of vectors, and the second holds because χ_v is orthogonal to every $\overline{\chi}_{S_\ell}$ with $\ell \neq j$. By the triangle inequality we have

$$\left\| x^{(T)} - \frac{1}{\sqrt{\operatorname{vol}(S_j)}} \overline{\chi}_{S_j} \right\| \leq \left\| x^{(T)} - \underline{\mathcal{I}}\chi_v \right\| + \left\| \underline{\mathcal{I}}\chi_v - \frac{1}{\sqrt{\operatorname{vol}(S_j)}} \overline{\chi}_{S_j} \right\|. \tag{6.12}$$

By Lemma 6.4 we have

$$\left\| x^{(T)} - \underline{\mathcal{I}}\chi_v \right\| = O\left(\frac{\log n \cdot \|\underline{\mathcal{I}}\chi_v\|}{\Upsilon_G(k)} + n^{-c} \right). \tag{6.13}$$

For any $v \in V$, let

$$\alpha_v \triangleq \sqrt{\frac{1}{d_v} \sum_{i=1}^{k} \left(f_i(v) - \widetilde{\chi}_i(v)\right)^2}.$$

Then it holds that

$$\begin{aligned}
\|\underline{\mathcal{I}}\chi_v\|^2 &= \sum_{i=1}^{k} \langle \chi_v, f_i \rangle^2 \\
&= \sum_{i=1}^{k} \langle \chi_v, \widetilde{\chi}_i - (\widetilde{\chi}_i - f_i) \rangle^2 \\
&= \sum_{i=1}^{k} \left(\langle \chi_v, \widetilde{\chi}_i \rangle - \langle \chi_v, \widetilde{\chi}_i - f_i \rangle \right)^2 \\
&\leq \sum_{i=1}^{k} 2 \left(\langle \chi_v, \widetilde{\chi}_i \rangle^2 + \langle \chi_v, \widetilde{\chi}_i - f_i \rangle^2 \right) && (6.14) \\
&= \frac{2}{\operatorname{vol}(S_i)} + 2 \sum_{i=1}^{k} \langle \chi_v, \widetilde{\chi}_i - f_i \rangle^2 && (6.15) \\
&\leq \frac{2}{\operatorname{vol}(S_i)} + 2\alpha_v^2 && (6.16)
\end{aligned}$$

where (6.14) follows from the inequality $(a-b)^2 \leq 2(a^2+b^2)$, (6.15) follows from (6.11), and (6.16) follows from the definition of α_v. Hence, it holds that

$$\|\underline{\mathcal{I}}\chi_v\| = O\left(\frac{1}{\sqrt{\operatorname{vol}(S_i)}} + \alpha_v \right). \tag{6.17}$$

For the second term in the right hand side of (6.12), by (6.11) and the triangle inequality,

$$\begin{aligned}
&\left\| \underline{\mathcal{I}}\chi_v - \frac{1}{\sqrt{\mathrm{vol}(S_j)}}\overline{\chi}_{S_j} \right\| \\
&= \left\| \sum_{i=1}^{k} \langle \chi_v, f_i \rangle f_i - \sum_{i=1}^{k} \langle \chi_v, f_i \rangle \widetilde{\chi}_i + \sum_{i=1}^{k} \langle \chi_v, f_i \rangle \widetilde{\chi}_i - \sum_{i=1}^{k} \langle \chi_v, \widetilde{\chi}_i \rangle \widetilde{\chi}_i \right\| \\
&\leq \left\| \sum_{i=1}^{k} \langle \chi_v, f_i \rangle f_i - \sum_{i=1}^{k} \langle \chi_v, f_i \rangle \widetilde{\chi}_i \right\| + \left\| \sum_{i=1}^{k} \langle \chi_v, f_i \rangle \widetilde{\chi}_i - \sum_{i=1}^{k} \langle \chi_v, \widetilde{\chi}_i \rangle \widetilde{\chi}_i \right\| \qquad (6.18)
\end{aligned}$$

Let's analyse the two terms in (6.18) separately. For the first term, it holds that

$$\begin{aligned}
\left\| \sum_{i=1}^{k} \langle \chi_v, f_i \rangle f_i - \sum_{i=1}^{k} \langle \chi_v, f_i \rangle \widetilde{\chi}_i \right\| &\leq \sum_{i=1}^{k} |\langle \chi_v, f_i \rangle| \, \| f_i - \widetilde{\chi}_i \| \\
&\leq O\left(\sqrt{\frac{k^3}{\Upsilon}} \right) \cdot \sum_{i=1}^{k} |\langle \chi_v, f_i \rangle| \qquad (6.19) \\
&\leq \sqrt{\frac{k^4}{\Upsilon_G(k)}} \, \| \underline{\mathcal{I}}\chi_v \| \\
&= O\left(\sqrt{\frac{k^4}{\Upsilon_G(k) \beta \, \mathrm{vol}(V)}} + \alpha_v \right) \qquad (6.20)
\end{aligned}$$

where the first inequality follows from the triangle inequality, the second by Lemma 6.5 and the Cauchy-Schwarz inequality, and the last inequality follows by (6.17). For the second term of (6.18), we have

$$\begin{aligned}
\left\| \sum_{i=1}^{k} \left(\langle \chi_v, f_i \rangle - \langle \chi_v, \widetilde{\chi}_i \rangle \right) \widetilde{\chi}_i \right\| = \left\| \sum_{i=1}^{k} \frac{1}{\sqrt{d_v}} \left(f_i(v) - \widetilde{\chi}_i(v) \right) \widetilde{\chi}_i \right\| &= \sqrt{\sum_{i=1}^{k} \frac{1}{d_v} \left(f_i(v) - \widetilde{\chi}_i(v) \right)^2} \\
&= \alpha_v, \qquad (6.21)
\end{aligned}$$

where the second equality follows from the orthonormality of $\{\widetilde{\chi}_i\}_i$. Putting all these inequalities above together, we have

$$\left\| x^{(T)} - \frac{1}{\sqrt{\mathrm{vol}(S_j)}}\overline{\chi}_{S_j} \right\| = O\left(\frac{\log n}{\Upsilon_G(k) \cdot \sqrt{\beta \, \mathrm{vol}(V)}} + n^{-c} + \sqrt{\frac{k^4}{\Upsilon_G(k) \beta \, \mathrm{vol}(V)}} + \alpha_v \right). \qquad (6.22)$$

Let's define the set

$$B \triangleq \left\{ v | \alpha_v > \sqrt{\frac{Ck^4 \log(1/\beta)}{\Upsilon_G(k)\beta \operatorname{vol}(V)}} \right\},$$

for some constant C.

Let us look at $\operatorname{vol}(B)$ now. By Lemma 6.5 we know that

$$\sum_{v \in V} d_v \alpha_v^2 = \sum_{i=1}^{k} \sum_{v \in V} \left(f_i(v) - \widetilde{\chi}_i(v)\right)^2 = \sum_{i=1}^{k} \left\| f_i - \widetilde{\chi}_i(v) \right\|^2 \leq \frac{k^4}{\Upsilon_G(k)}.$$

It follows that

$$\operatorname{vol}(B) \leq \frac{k^4}{\Upsilon_G(k)} \cdot \frac{\Upsilon_G(k)\beta \operatorname{vol}(V)}{Ck^4 \log(1/\beta)} = \frac{\beta \operatorname{vol}(V)}{C \log(1/\beta)}. \tag{6.23}$$

By (6.22), and the definition of B, we have that for all $v \notin B$, by setting $x^{(0)} = \chi_v$ we have that

$$\begin{aligned} \left\| x^{(T)} - \frac{1}{\sqrt{\operatorname{vol}(S_j)}} \overline{\chi}_{S_j} \right\| &= O\left(\frac{\log n}{\Upsilon_G(k) \cdot \sqrt{\beta \operatorname{vol}(V)}} + \sqrt{\frac{k^4}{\Upsilon \beta \operatorname{vol}(V)}} + \sqrt{\frac{Ck^4 \log(1/\beta)}{\Upsilon_G(k)\beta \operatorname{vol}(V)}} \right) \\ &= O\left(\sqrt{\frac{k^4 \log(1/\beta)}{\Upsilon_G(k)\beta \operatorname{vol}(V)}} \right), \end{aligned}$$

since it holds by (6.7) that $\log n / \Upsilon = o(1)$. This implies that $V \setminus B \subseteq A$, yielding the claimed statement. □

So far we have analysed the case for a single initial vector. To identify all the k clusters simultaneously in T rounds, we repeat this process multiple times with carefully chosen initial vectors. In particular, we need to ensure that we start the averaging procedure from at least one vertex in each cluster. This is the reason for us to introduce the seeding procedure. By setting the probability $\bar{s} \cdot d_v / \operatorname{vol}(V)$ for every vertex v to be active, it is easy to prove that with constant probability there is at least an active vertex in each cluster.

Our analysis for the query procedure is based on the relation between our averaging procedure and lazy random walks: since any single random walk gets well mixed inside a cluster after T steps, we expect that the states of the vertices inside a cluster are similar. Namely, for the cluster S_j to which the initial vertex of the ith vector belongs to, on a regular graph we expect that $x^{(T,i)}(v) \approx 1/\operatorname{vol}(S_j) \geq 1/(\beta \operatorname{vol}(V))$ for most

$v \in S_j$ and $x^{(T,i)}(v) \approx 0$ otherwise. Therefore, based on (6.10), vertices from the same cluster will choose the same label, while vertices from different clusters will choose different ones.

Proof of Theorem 6.2. For each vertex v, the probability that we start the averaging procedure with an initial vector χ_v is equal to $\bar{s} \cdot d_v/\mathrm{vol}(V)$, where $\bar{s} = \Theta\left(\frac{1}{\beta}\log\frac{1}{\beta}\right)$. Hence, the probability that there exists a j such that no vertex from S_j is chosen as initial vertex is at most

$$\prod_{v\in S_j}\left(1-\frac{\bar{s}\cdot d_v}{\mathrm{vol}(V)}\right) \leq \prod_{v\in S_j} \mathrm{e}^{-\bar{s}\cdot d_v/\mathrm{vol}(V)} = \mathrm{e}^{-\bar{s}\sum_{v\in S_j} d_v/\,\mathrm{vol}(V)} \leq \frac{1}{200k},$$

where we used the inequality $1-x \leq \mathrm{e}^{-x}$ for $x \leq 1$, the assumption on the size of the clusters, i.e., $\mathrm{vol}(S_j) \geq \beta\,\mathrm{vol}(V)$, and the trivial fact that $\beta \leq 1/k$. As a consequence, with probability greater than $1/200$, for each cluster S_j, at least one vertex $v \in S_j$ is chosen as a starting vertex of the averaging process.

Next, we bound the probability that all the starting vertices belong to the set A defined in Lemma 6.6. By the algorithm description, the actual number of active vertices s satisfies $\mathbb{E}[s] = \bar{s}$. Therefore, it holds with probability $1 - O(1)$ that $s = O\left(\frac{1}{\beta}\log\frac{1}{\beta}\right)$. We assume that this event occurs in the rest of the proof. Let $v_1, \ldots, v_s$ be the starting vertices. By Lemma 6.6, the probability that there exists a starting vertex v_i not belonging to A is at most

$$\begin{aligned}\mathbb{P}[\text{there is some starting vertex } v_i \notin A] &\leq \frac{O(\bar{s})\cdot(\mathrm{vol}(V)-\mathrm{vol}(A))}{\mathrm{vol}(V)} \leq \frac{O(\bar{s})\cdot\beta}{C\log(1/\beta)}\\ &\leq \frac{1}{200}.\end{aligned}$$

Hence, with probability $1-O(1)$ every starting vertex belongs to A. For the rest of the proof we assume this is the case. For any vertex v, let $\mathcal{S}(v)$ be the cluster v belongs to. Then, by the definition of the set A, it holds for any starting vertex v_i that

$$\left\| x^{(T,i)} - \frac{\overline{\chi}_{\mathcal{S}(v_i)}}{\sqrt{\mathrm{vol}(\mathcal{S}(v_i))}} \right\| = O\left(\sqrt{\frac{k^4\log(1/\beta)}{\Upsilon_G(k)\beta\,\mathrm{vol}(V)}}\right). \tag{6.24}$$

Observe that a vertex v is misclassified by the algorithm only if there exists $i \in \{1,\ldots,s\}$ such that

$$\left|\frac{x^{(T,i)}(v)}{\sqrt{d_v}} - \frac{\overline{\chi}_{\mathcal{S}(v_i)}(v)}{\sqrt{d_v\cdot\mathrm{vol}(\mathcal{S}(v_i))}}\right|^2 > \frac{1}{4\beta^2\,\mathrm{vol}(V)^2}. \tag{6.25}$$

Then, by (6.24) the total volume of misclassified vertices is at most

$$\begin{aligned} &O(\bar{s}) \cdot \left\| x^{(T,i)} - \frac{1}{\sqrt{\mathrm{vol}(\mathcal{S}(v_i))}} \overline{\chi}_{\mathcal{S}(v_i)} \right\|^2 \cdot 4\beta^2 \,\mathrm{vol}(V)^2 \\ &= O\left(\frac{1}{\beta} \log \frac{1}{\beta} \cdot \frac{k^4}{\Upsilon_G(k)} \cdot \frac{\log(1/\beta)}{\beta \,\mathrm{vol}(V)} \cdot \beta^2 \,\mathrm{vol}(V)^2 \right) \\ &= O\left(\frac{k^4}{\Upsilon_G(k)} \log^2 \frac{1}{\beta} \right) \mathrm{vol}(V). \end{aligned}$$

Combining this with the assumption (6.7) proves the first statement. The second statement follows by the fact that the total communication among all vertices in each round is $O(m \cdot (1/\beta) \log(1/\beta))$ words. □

6.3 Experiments

The aim of this section is to present experimental results on our sparsification procedure and to show that, on realistic examples, Algorithm 4 is able to preserve the cluster-structure of graphs. In particular, we want to show that applying Spectral Clustering to a graph G and to its sparsification H produces almost identical results. For this reason, our focus will be restricted on graphs for which Spectral Clustering performs well. We will test our sparsification procedure on similarity graphs obtained from artificial distribution of points in the plane and from real-world images. This choice stems from two facts: (1) Spectral Clustering is well-known to perform well on image segmentation tasks [60]; (2) similarity graphs are inherently dense, a prerequisite for our sparsification procedure to have any meaning at all.

We use two functions to compare the quality of the clustering produced on the original graph G and its sparsification H:

1. For synthetic datasets for which the underlying ground-truth clustering is known, the quality of a clustering algorithm is measured by the ratio of misclassified vertices, i.e.,
$$\mathsf{err}(A_1, \ldots, A_k) \triangleq \frac{1}{n} \cdot \sum_{i=1}^{k} |\{v \in A_i : v \notin S_i\}|,$$
where $\{S_1, \cdots, S_k\}$ is the underlying ground-truth clustering and $\{A_1, \ldots, A_k\}$ is the one returned by the clustering algorithm.

2. For datasets for which a ground-truth clustering is not well-defined, the quality of a clustering is measured by the *normalised cut value* defined by

$$\mathsf{ncut}(A_1, \ldots, A_k) \triangleq \sum_{i=1}^{k} \frac{w(A_i, V \setminus A_i)}{\mathrm{vol}(A_i)},$$

which is just a variant of the k-way expansion (2.9) but more commonly used in the machine learning literature [60, 74].

All the experiments are conducted with Matlab and we use an implementation of the classical Spectral Clustering algorithm described in [50].

6.3.1 Datasets

We test the algorithms in the following three synthetic and real-world datasets, which are visualised in Figure 6.2.

- `Twomoons`: this dataset consists of n points in $\mathbb{R}^2$, where n is chosen between $1,000$ and $15,000$. We consider each point to be a vertex. For any two vertices u, v, we add an edge with weight $w(u, v) = \exp(-\|u - v\|^2/2\sigma^2)$, where $\sigma = 0.1$.

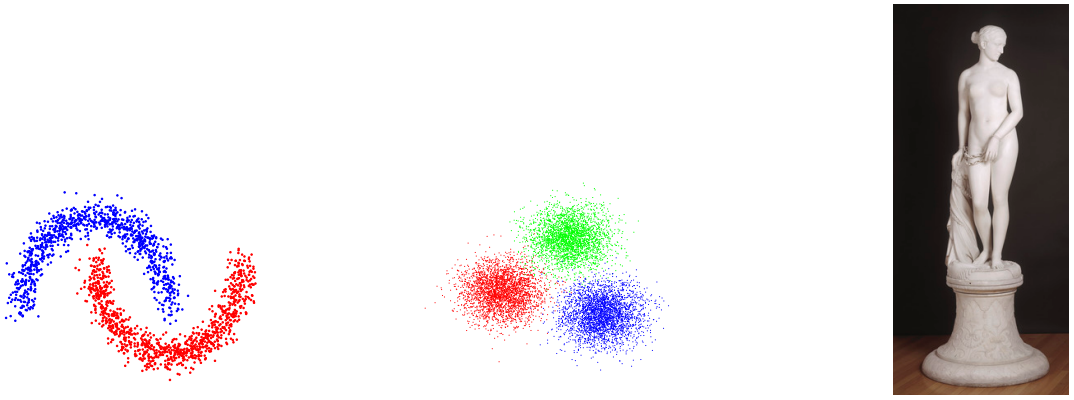
- `Gaussians`: this dataset consists of n points in $\mathbb{R}^2$, where n is chosen between $1,000$ and $15,000$. Each point is sampled from a uniform mixture of 3 isotropic Gaussians of variance 0.04. The similarity graph is constructed in the same way as `Twomoons`, and we set $\sigma = 1$ here.

- `Sculpture`: we use a 73×160 version of a photo of *The Greek Slave*[1] where each pixel is viewed as a vertex. To construct a similarity graph, we map each pixel to a point in $\mathbb{R}^5$, i.e., (x, y, r, g, b), where the latter three coordinates are the RGB values. For any two vertices u, v, we put an edge between u and v with weight $w(u, v) = \exp(-\|u - v\|^2/2\sigma^2)$, where $\sigma = 20$. This results in a graph with about $11,000$ vertices and $k = 3$ clusters.

These datasets are essentially the ones used in [15], which studies the effects of spectral sparsification on clustering. This makes it possible to easily compare our results with the state-of-the-art. The choice of σ varies for different datasets, since they have in general different intra-cluster variance. There are several heuristics to

[1] http://artgallery.yale.edu/collections/objects/14794

choose the "correct" value of σ (see, e.g., the classical reference [50]). In our case the value of σ is chosen so that the spectral gap $|\lambda_{k+1} - \lambda_k|$ of the original similarity graph is large. This ensures that the clusters in the graph are well-defined, and Spectral Clustering outputs a meaningful clustering.



(a) `Twomoons` (b) `Gaussians` (c) `Sculpture`

Figure 6.2: Visualisation of the datasets used in our experiments.

6.3.2 Experimental results

We test the performance of our algorithm on the three datasets. Notice that the sampling probability of the edges in our first algorithm involves the factor $\tau \geq C/\lambda_{k+1}$. To find a desired value of τ, we use the following doubling method: starting with $\tau = 0.1$, we double the value of τ each time, until the spectral gap $|\lambda_{k+1} - \lambda_k|$ of the resulting matrices doesn't change significantly. Remarkably, for all the datasets considered in this section, $\tau = 1.6$ always suffices for our purposes. Notice that this method will only increase the time complexity of our algorithm by at most a poly-logarithmic factor of n.

For the `Twomoons` and `Gaussians` datasets, for all the tested graphs with size ranging from $1,000$ to $15,000$ points, our sparsified graphs require only about $0.14\% \sim 3.13\%$ of the total edges. The error ratios of Spectral Clustering on the original datasets and our sparsified graphs are listed respectively as err1 and err2, and are always very close. See Table 6.1 and Table 6.2 for details.

The `Sculpture` dataset corresponds to a similarity graph of $n = 11,680$ vertices and about 68 million edges. We run Spectral Clustering on both the input graph and our sparsified one, and compute the normalised cut values of each clustering in the original input graph. By setting $\tau = 1.6$, our algorithm samples only 0.37% of the

Table 6.1: Experimental results for the `Twomoons` dataset, where $\tau = 0.8$.

n	# edges (%)	err1 (%)	err2 (%)
1,000	1.56	0.900	0.600
2,000	0.86	0.150	0.100
4,000	0.48	0.150	0.175
8,000	0.26	0.086	0.088
10,000	0.22	0.120	0.150
15,000	0.14	0.080	0.100

Table 6.2: Experimental results for the `Gaussians` dataset, where $\tau = 1.6$.

n	# edges (%)	err1 (%)	err2 (%)
1,000	3.13	1.700	1.900
2,000	1.75	1.350	1.650
4,000	0.96	0.400	0.417
8,000	0.66	0.128	0.140
10,000	0.42	0.113	0.119
15,000	0.29	0.125	0.148

edges (320,000) from the input graph. The normalised cut value of Spectral Clustering on the original dataset is 0.0938, while the normalised cut value of Spectral Clustering on our sparsified graph is 0.0935. The visualisation of the two clustering results are almost identical, as shown in Figure 6.3.

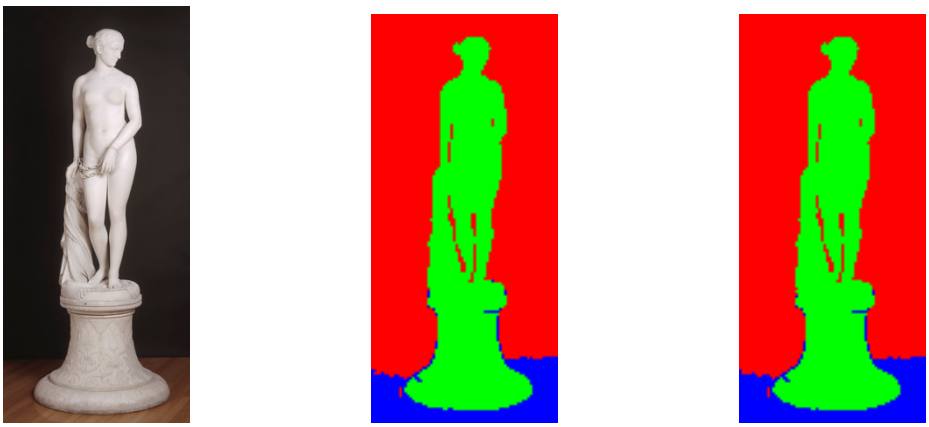


Figure 6.3: Visualisation of the results on the dataset `Sculpture`. The left-side picture is the original input dataset, the middle one is the outcome of Spectral Clustering on the original input dataset, while the right-side picture is the outcome of Spectral Clustering on our sparsified graph.

6.4 Open problems

In this chapter we have shown Spectral Clustering can be efficiently implemented in the distributed setting. There are, however, slightly stronger assumptions that need to be satisfied for the distributed implementation of Spectral Clustering to have comparable results with the classical sequential algorithm. Among these assumptions, probably the most critical one is the need to know a good estimate of λ_{k+1}. Even though the results of Section 6.3 suggests that in many practical scenarios such value can be guessed, it remains an interesting problem to see if λ_{k+1} can be approximated efficiently in the distributed setting. We remark that a partial solution to this problem has been proposed in [11]: their algorithm only requires a (loose) lower bound on λ_{k+1}, which as in our case determines the number of rounds needed. In contrast, the algorithm presented in this chapter requires at least an upper bound to λ_k as well, since the algorithm must necessarily stop before $1/\lambda_k$ rounds.

The other main drawback of the distributed implementation compared to the sequential one is the requirement that clusters need to be balanced in size. In Chapter 4 we have seen Spectral Clustering is able to detect even very small clusters. In the distributed case this appears much more challenging. Approaches based on random walks are unlikely to work, since it's very difficult to start a random walk inside one of these small clusters: suppose we start a random walk from a vertex chosen uniformly at random, then, with high probability such vertex won't belong to one of the small clusters. We remark that the current analysis of our algorithm doesn't give any guarantee on the clustering produced when even a single cluster is small. We do not rule out the possibility that a more careful analysis would show that, in the presence of small clusters, our algorithm still outputs a good approximation of the large ones.

References

[1] Abbe, E. (2017). Community detection and stochastic block models: recent developments. *ArXiv e-print 1703.10146*.

[2] Abbe, E. and Sandon, C. (2015). Community detection in general stochastic block models: Fundamental limits and efficient algorithms for recovery. In *The 56th Annual Symposium on the Foundations of Computer Science*, pages 670–688.

[3] Achlioptas, D. (2003). Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *Journal of computer and System Sciences*, 66(4):671–687.

[4] Ackerman, M. and Ben-David, S. (2009). Clusterability: A theoretical study. In *The 12th International Conference on Artificial Intelligence and Statistics*, pages 1–8.

[5] Allen Zhu, Z., Lattanzi, S., and Mirrokni, V. (2013). A local algorithm for finding well-connected clusters. In *The 30th International Conference on Machine Learning*, pages 396–404.

[6] Alon, N. and Milman, V. D. (1985). λ_1, isoperimetric inequalities for graphs, and superconcentrators. *Journal of Combinatorial Theory, Series B*, 38(1):73–88.

[7] Arora, S., Barak, B., and Steurer, D. (2010). Subexponential algorithms for unique games and related problems. In *The 51st Annual Symposium on the Foundations of Computer Science*, pages 563–572.

[8] Arora, S., Rao, S., and Vazirani, U. (2009). Expander flows, geometric embeddings and graph partitioning. *Journal of the ACM*, 56(2):5.

[9] Arthur, D. and Vassilvitskii, S. (2007). k-means++: The advantages of careful seeding. In *The 18th Annual Symposium on Discrete Algorithms*, pages 1027–1035.

[10] Batson, J., Spielman, D. A., Srivastava, N., and Teng, S.-H. (2013). Spectral sparsification of graphs: Theory and algorithms. *Communications of the ACM*, 56(8):87–94.

[11] Becchetti, L., Clementi, A., Natale, E., Pasquale, F., and Trevisan, L. (2017). Find your place: Simple distributed algorithms for community detection. In *The 28th Annual Symposium on Discrete Algorithms*, pages 940–959.

[12] Boppana, R. B. (1987). Eigenvalues and graph bisection: An average-case analysis. In *The 28th Annual Symposium on the Foundations of Computer Science*, pages 280–285.

[13] Bădoiu, M., Har-Peled, S., and Indyk, P. (2002). Approximate clustering via core-sets. In *The 34th Symposium on Theory of Computing*, pages 250–257.

[14] Cheeger, J. (1969). A lower bound for the smallest eigenvalue of the Laplacian. In *Proceedings of the Princeton conference in honor of Professor S. Bochner*.

[15] Chen, J., Sun, H., Woodruff, D., and Zhang, Q. (2016). Communication-optimal distributed clustering. In *The 29th Conference on Neural Information Processing Systems*, pages 3720–3728.

[16] Chin, P., Rao, A., and Vu, V. (2015). Stochastic block model and community detection in sparse graphs: A spectral algorithm with optimal rate of recovery. In *Proceedings of The 28th Conference on Learning Theory*, pages 391–423.

[17] Chung, F. R. K. (1997). *Spectral graph theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. American Mathematical Society.

[18] Coja-Oghlan, A. (2010). Graph partitioning via adaptive spectral techniques. *Combinatorics, Probability and Computing*, 19(02):227–284.

[19] Davis, C. and Kahan, W. M. (1970). The rotation of eigenvectors by a perturbation. III. *SIAM Journal on Numerical Analysis*, 7(1):1–46.

[20] De La Vega, W. F., Karpinski, M., Kenyon, C., and Rabani, Y. (2003). Approximation schemes for clustering problems. In *The 35th Symposium on Theory of Computing*, pages 50–58.

[21] Decelle, A., Krzakala, F., Moore, C., and Zdeborová, L. (2011). Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, 84(6):066106.

[22] Dodziuk, J. (1984). Difference equations, isoperimetric inequality and transience of certain random walks. *Transactions of the American Mathematical Society*, 284(2):787–794.

[23] Dubhashi, D. P. and Panconesi, A. (2009). *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press.

[24] Fiedler, M. (1973). Algebraic connectivity of graphs. *Czechoslovak mathematical journal*, 23(2):298–305.

[25] Fiedler, M. (1975). A property of eigenvectors of nonnegative symmetric matrices and its application to graph theory. *Czechoslovak Mathematical Journal*, 25(4):619–633.

[26] Fortunato, S. (2010). Community detection in graphs. *Physics reports*, 486(3):75–174.

[27] Guattery, S. and Miller, G. L. (1998). On the quality of spectral separators. *SIAM Journal on Matrix Analysis and Applications*, 19(3):701–719.

[28] Holland, P. W., Laskey, K. B., and Leinhardt, S. (1983). Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137.

[29] Horn, R. A. and Johnson, C. R. (2012). *Matrix analysis*. Cambridge University Press.

[30] Indyk, P. and Motwani, R. (1998). Approximate nearest neighbors: towards removing the curse of dimensionality. In *The 30th Symposium on Theory of Computing*, pages 604–613.

[31] Johnson, W. B. and Lindenstrauss, J. (1984). Extensions of Lipschitz mappings into a Hilbert space. *Contemporary mathematics*, 26(189-206):1.

[32] Kannan, R., Vempala, S., and Vetta, A. (2004). On clusterings: Good, bad and spectral. *Journal of the ACM*, 51(3):497–515.

[33] Kanungo, T., Mount, D. M., Netanyahu, N. S., Piatko, C. D., Silverman, R., and Wu, A. Y. (2002). A local search approximation algorithm for k-means clustering. In *Proceedings of The 18th Annual Symposium on Computational Geometry*, pages 10–18.

[34] Kelner, J. A., Lee, J. R., Price, G. N., and Teng, S.-H. (2011). Metric uniformization and spectral bounds for graphs. *Geometric and Functional Analysis*, 21(5):1117–1143.

[35] Kempe, D. and McSherry, F. (2004). A decentralized algorithm for spectral analysis. In *The 36th Symposium on Theory of Computing*, pages 561–568.

[36] Kirchhoff, G. (1847). Ueber die auflösung der gleichungen, auf welche man bei der untersuchung der linearen vertheilung galvanischer ströme geführt wird. *Annalen der Physik*, 148(12):497–508.

[37] Kumar, A., Sabharwal, Y., and Sen, S. (2004). A simple linear time $(1+\varepsilon)$-approximation algorithm for k-means clustering in any dimensions. In *The 45th Annual Symposium on the Foundations of Computer Science*, pages 454–462.

[38] Kwok, T. C., Lau, L. C., Lee, Y. T., Oveis Gharan, S., and Trevisan, L. (2013). Improved Cheeger's inequality: analysis of spectral partitioning algorithms through higher order spectral gap. In *The 45th Symposium on Theory of Computing*, pages 11–20.

[39] Lee, J. R., Gharan, S. O., and Trevisan, L. (2014). Multiway spectral partitioning and higher-order Cheeger inequalities. *Journal of the ACM*, 61(6):37.

[40] Lei, J., Rinaldo, A., et al. (2015). Consistency of spectral clustering in stochastic block models. *The Annals of Statistics*, 43(1):215–237.

[41] Lloyd, S. (1982). Least squares quantization in PCM. *IEEE transactions on information theory*, 28(2):129–137.

[42] Louis, A., Raghavendra, P., Tetali, P., and Vempala, S. (2012). Many sparse cuts via higher eigenvalues. In *The 44th Symposium on Theory of Computing*, pages 1131–1140.

[43] Mahajan, M., Nimbhorkar, P., and Varadarajan, K. (2009). The planar k-means problem is NP-hard. In *International Workshop on Algorithms and Computation*, pages 274–285.

[44] Mahoney, M. W. and Orecchia, L. (2011). Implementing regularization implicitly via approximate eigenvector computation. In *The 28th International Conference on Machine Learning*.

[45] Massoulié, L. (2014). Community detection thresholds and the weak Ramanujan property. In *The 46th Symposium on Theory of Computing*, pages 694–703.

[46] Matoušek, J. (2000). On approximate geometric k-clustering. *Discrete & Computational Geometry*, 24(1):61–84.

[47] Matula, D. W. and Shahrokhi, F. (1990). Sparsest cuts and bottlenecks in graphs. *Discrete Applied Mathematics*, 27(1-2):113–123.

[48] McSherry, F. (2001). Spectral partitioning of random graphs. In *The 42nd Annual Symposium on the Foundations of Computer Science*, pages 529–537.

[49] Mossel, E., Neeman, J., and Sly, A. (2015). Reconstruction and estimation in the planted partition model. *Probability Theory and Related Fields*, 162(3-4):431–461.

[50] Ng, A. Y., Jordan, M. I., and Weiss, Y. (2001). On spectral clustering: Analysis and an algorithm. In *The 14th Conference on Neural Information Processing Systems*, number 2, pages 849–856.

[51] Orecchia, L., Sachdeva, S., and Vishnoi, N. K. (2012). Approximating the exponential, the Lanczos method and an $\widetilde{O}(m)$-time spectral algorithm for balanced separator. In *The 44th Symposium on Theory of Computing*, pages 1141–1160.

[52] Ostrovsky, R., Rabani, Y., Schulman, L. J., and Swamy, C. (2012). The effectiveness of lloyd-type methods for the k-means problem. *Journal of the ACM*, 59(6):28.

[53] Oveis Gharan, S. and Trevisan, L. (2014). Partitioning into expanders. In *The 25th Annual Symposium on Discrete Algorithms*, pages 1256–1266.

[54] Peleg, D. (2000). *Distributed computing: a locality-sensitive approach*. SIAM.

[55] Peng, R., Sun, H., and Zanetti, L. (2015). Partitioning well-clustered graphs: Spectral clustering works! In *The 28th Annual Conference on Learning Theory*, pages 1423–1455.

[56] Peng, R., Sun, H., and Zanetti, L. (2017). Partitioning well-clustered graphs: Spectral clustering works! *SIAM Journal on Computing*, 46(2):710–743.

[57] Qin, T. and Rohe, K. (2013). Regularized spectral clustering under the degree-corrected stochastic blockmodel. In *The 26th Conference on Neural Information Processing Systems*, pages 3120–3128.

[58] Rohe, K., Chatterjee, S., and Yu, B. (2011). Spectral clustering and the high-dimensional stochastic blockmodel. *The Annals of Statistics*, pages 1878–1915.

[59] Saloff-Coste, L. (1997). Lectures on finite markov chains. In *Lectures on probability theory and statistics*, pages 301–413. Springer.

[60] Shi, J. and Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905.

[61] Šıma, J. and Schaeffer, S. E. (2006). On the NP-completeness of some graph cluster measures. In *Proceedings of the 32nd International Conference on Current Trends in Theory and Practice of Computer Science*, volume 3831, pages 530–537.

[62] Sinclair, A. and Jerrum, M. (1989). Approximate counting, uniform generation and rapidly mixing Markov chains. *Information and Computation*, 82(1):93–133.

[63] Spielman, D. A. and Srivastava, N. (2011). Graph sparsification by effective resistances. *SIAM Journal on Computing*, 40(6):1913–1926.

[64] Spielman, D. A. and Teng, S.-H. (2011). Spectral sparsification of graphs. *SIAM Journal on Computing*, 40(4):981–1025.

[65] Spielman, D. A. and Teng, S.-H. (2013). A local clustering algorithm for massive graphs and its application to nearly linear time graph partitioning. *SIAM Journal on Computing*, 42(1):1–26.

[66] Spielman, D. A. and Teng, S.-H. (2014). Nearly linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. *SIAM Journal on Matrix Analysis and Applications*, 35(3):835–885.

[67] Sun, H. and Zanetti, L. (2017a). Distributed graph clustering and sparsification. Manuscript in submission. Invited to the SPAA'17 special issue of *ACM Transactions on Parallel Computing*.

[68] Sun, H. and Zanetti, L. (2017b). Distributed graph clustering by load balancing. In *The 29th Annual Symposium on Parallel Algorithms and Architectures*, pages 163–171.

[69] Teng, S.-H. (2010). The Laplacian paradigm: Emerging algorithms for massive graphs. In *International Conference on Theory and Applications of Models of Computation*, pages 2–14.

[70] Tolliver, D. A. and Miller, G. L. (2006). Graph partitioning by spectral rounding: Applications in image segmentation and clustering. In *Proceedings of the 2006 IEEE*

Computer Society Conference on Computer Vision and Pattern Recognition, pages 1053–1060.

[71] Trevisan, L. (2013). Lecture notes for graph partitioning and expanders.

[72] Tropp, J. A. et al. (2015). An introduction to matrix concentration inequalities. *Foundations and Trends in Machine Learning*, 8(1-2):1–230.

[73] Verma, D. and Meila, M. (2003). Comparison of spectral clustering methods. In *The 16th Conference on Neural Information Processing Systems*.

[74] Von Luxburg, U. (2007). A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416.